UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE MATEMÁTICA

Proyecto de grado

FUNDAMENTO ALGEBRAICO DEL ALGORITMO AES (Estándar de Encriptación Avanzada) PARA LA SEGURIDAD DE DATOS

PRESENTADO PARA OPTAR AL GRADO DE LICENCIADO EN MATEMÁTICA

Postulante: Julio Rodrigo Zabala Tarifa

Tutor: Ph.D. Ramiro Hernán Lafuente Rodriguez

La Paz - Bolivia

2019

Dedicado a las futuras mentes matemáticas,
con la esperanza de estarles legando
un camino más allanado.

Agradecimientos

Un largo camino ha culminado.

Durante su recorrido, a mi lado han estado muchas y grandes personas aportando a mi causa ya sea con su apoyo moral o con sus conocimientos y experiencia, sea pues con todos ellos un entrañable y perenne reconocimiento.

No obstante estas líneas están consagradas principalmente a ofrecer infinita e inconmensurable gratitud a dos personas divinas y hermosas que Dios ha puesto en mi vida como una gran bendición…

Mi adorada madre *Felicidad*:

Jamás dejaste de creer en mi; tu fe fue, es y será inquebrantable, que gloriosa enseñanza; siempre has estado a mi lado en la trinchera hombro con hombro y siempre preparada para la lucha contra toda adversidad, porque la implacable e inagotable fuerza de tu corazón valeroso y abnegado proviene del infinito amor existente en él …

Y mi amada esposa *Fabiola*:

Tu guía en este tramo final ha sido dura y severa; pero atinada, vaya verdad oculta en un sabio proceder que finalmente ha dado sus frutos pues he aprendido de ti sobre como el mayor poder humano emana del amor y la inmensa pasión que se le pone a lo que se hace.

Son estos los caudales indómitos de energía y vida de los que mi alma ha bebido y se ha hecho aún mas fuerte… Gracias a ambas por ser a la vez mi escudo, mi armadura y mi espada.

Resumen

Se presentan los fundamentos algebraicos así como también varias propiedades del algoritmo AES.

Comenzando por el estudio de la construcción del campo finito en el cual se basa el cifrado denominado el campo de Rijndael y cuyos elementos serán plausiblemente identificados con entes informáticos tales como los bytes y los bits, además de sus modos estándar de operar. Se destaca la combinación de funciones lineales, que en campos finitos no son otra cosa que los llamados polinomios linealizados, con funciones no lineales lo cual permitirá la estructuración de la función de ronda que cumplirá con los requerimientos planteados por la estrategia del rastro amplio, cuyo tratamiento comprende el enfoque cripto-analítico presente en varios pasajes del presente trabajo, tocando tópicos como ataques de tipo lineal, diferencial y de interpolación. Además se consideran las diferentes representaciones del algoritmo provenientes de diferentes formas de manejo de los elementos del campo de Rijndael y la estructura algebraica subyacente. Finalmente es llevada a cabo la descripción polinomial del algoritmo AES consistente en el cálculo de los coeficientes de la expresión polinómica de la función de ronda.

Índice general

1. Aspectos Generales **1**
1.1. Prólogo 1
1.2. Introducción 3
1.3. Antecedentes 4
1.4. Objetivo del Trabajo 5
1.5. Planteamiento del problema 5
1.6. Alcance 6
1.7. Metodología 6
1.8. Medios 6

I Fundamentos Matemáticos **7**

2. Anillos de Polinomios **8**
2.1. Anillos de Polinomios Univariantes 8
2.2. Anillos de Polinomios Multivariantes 10
2.3. Álgebras 12

3. Propiedades de Campos Finitos **13**
3.1. Campos Finitos y Subcampos 13
3.2. Construcción Explícita de Campos Finitos 14
3.3. Representaciones para un Campo Finito 15
3.4. Funciones en un Campo Finito 17
3.5. Funciones Booleanas 19

II Descripción del Algoritmo AES y sus Especificaciones **21**

4. Estándar de Encriptación Avanzada AES **22**
4.1. Notación 23

4.1.1. Entradas y Salidas 23
4.1.2. Bytes 24
4.1.3. El Estado 25
4.2. El Sustento Matemático 26
4.2.1. Polinomios con Coeficientes en GF (2^8) 26
4.3. Especificaciones del Algoritmo 28
4.4. Las Etapas 29
4.4.1. **`SubBytes`** 29
4.4.2. **`ShiftRows`** 30
4.4.3. **`MixColumns`** 31
4.4.4. **`AddRoundKey`** 32
4.4.5. Programa de Expansión de la Clave 32
4.4.6. Construcción de la S-box 33
4.5. Desencriptación 34
4.6. Consideraciones de Diseño 38
4.6.1. Estrato de Sustitución 38
4.6.2. Estrato de Difusión 39
4.6.3. Estrato de Incorporación de la Clave 39

5. Propiedades Algebraicas del AES **40**
5.1. Estructura de Ronda 40
5.1.1. La Operación **`SubBytes`** 41
5.1.2. Las Operaciones **`ShiftRows`** y **`MixColumns`** 41
5.1.3. Difusión Lineal Aumentada 42
5.1.4. Propiedades de la Difusión Lineal Aumentada 44
5.2. Representaciones Algebraicas 46
5.2.1. Representaciones del AES 48
5.3. Gran Sistema de Encriptación (BES) 48
5.3.1. Aplicación de Encaje del AES en el BES 48
5.3.2. Estructura del BES 49
5.3.3. Función de Ronda del BES 50
5.3.4. Encaje o Inmersión del AES en el BES 50
5.4. Otras Representaciones del AES 51
5.4.1. Cifrados Isomorfos 51
5.4.2. Representaciones Regulares 53
5.4.3. Representaciones Logarítmicas 55
5.5. Propiedades Inherentes a la Teoría de Grupos 55

5.5.1. Cifrados Iterados por Bloques 55
5.5.2. Estructuras de Ciclo 57
5.5.3. Grupo de Difusión de Bytes 58
5.5.4. Propiedades Geométricas 60

III Rijndael-GF 63

6. Análisis de la Propagación en Campos de Galois 64
6.1. Funciones sobre GF (2^n) 65
6.1.1. Propagación de la Diferencia 65
6.1.2. Correlación 66
6.1.3. Funciones que son Lineales sobre GF (2^n) 68
6.1.4. Funciones que son Lineales sobre GF (2) 69
6.2. Funciones sobre $(\text{GF}(2^n))^l$ 70
6.2.1. Propagación de la Diferencia 70
6.2.2. Correlación 70
6.2.3. Funciones que son Lineales sobre GF (2^n) 71
6.2.4. Funciones que son Lineales sobre GF (2) 71

7. Representaciones de GF (p^n) **73**
7.1. Representación cíclica de GF (p^n) 73
7.2. Representación Vectorial de GF (p^n) 73
7.3. Bases Duales 74

8. Funciones Booleanas y Funciones en GF (2^n) **76**
8.1. Diferencias en GF $(2)^n$ y GF (2^n) 76
8.2. Relación entre Patrones de Traza y Patrones de Selección 77
8.3. Relación entre Funciones Lineales en GF $(p)^n$ y en GF (p^n) 77
8.4. Ilustración 79

9. Rijndael-GF 82

10 Descripción Polinomial del Rijndael AES 84
10.1 Rijndael bajo el Enfoque de la Teoría de Anillos 84
10.2 Relación con la Descripción Estándar 87
10.2.1 AES-192 y AES-256 89
10.3 La Estructura de la S-box 89

11 Epílogo **94**
11.1 Conclusiones 94
11.2 Recomendaciones 94

Capítulo 1

Aspectos Generales

1.1. Prólogo

A lo largo de la historia siempre ha estado presente en la humanidad esa fascinación por mantener información lejos del alcance de otros. Reyes y generales han hecho uso de métodos básicos de criptografía para la comunicación con sus tropas, dejando al enemigo sin acceso a información militar sensible. De hecho, se tiene documentado que Julio Cesar utilizó un cifrado simple que ha sido bautizado con su nombre. Parece ser algo innato en el ser humano pues incluso los niños muestran la capacidad de implantar en el juego la comunicación de la información de manera encubierta por medio de señales, gestos o palabras claves, para ocultar su contenido de otro grupo o de una figura de autoridad. Como sociedad hemos evolucionado, por lo cual la necesidad de contar con métodos más sofisticados de protección de datos se ha incrementado, además vivimos en la era de la información a la mano, en cualquier momento o lugar y en cualquier cantidad, así pues, tal necesidad es más pronunciada que nunca. El orbe entero se encuentra conectado y por consiguiente la demanda de información y servicios electrónicos está creciendo vertiginosamente, generando una dependencia de los sistemas electrónicos en todo ámbito. Hoy por hoy el intercambio de información sensible es muy común y cotidiano, esto es patente en por ejemplo la banca por internet o las compras y subastas en línea. La protección de tales datos y su manejo electrónico son de crucial relevancia para nuestro estilo de vida. La seguridad en la transferencia digital de datos en el mundo contemporáneo se constituye también en un aspecto de vital importancia para la seguridad nacional de la gran mayoría de los países y ha generado en las últimas décadas un gran desarrollo de investigación matemática, principalmente en Álgebra. Las técnicas necesarias para la protección de la información están enmarcadas en el ámbito de la criptología. En la actualidad se considera la criptología como el estudio de la comunicación a través de canales no seguros y problemas relacionados; tal disciplina engloba a la criptografía, la ciencia del diseño de sistemas de encriptación y la criptología, la ciencia de los métodos de ataque y ruptura de los sistemas de encriptación, siendo ambas áreas claramente interdependientes entre ellas.

A menudo el término Teoría de Codificación es usado para describir la criptografía, sin embargo ésto puede confundirnos. La Teoría de Códigos trata de la representación de los datos de entrada de un sistema expresados en símbolos, por otros símbolos llamados códigos. Hay 3 aplicaciones básicas que cubre la Teoría de Códigos: comprensión, discreción y corrección de errores. Más aún en el último tiempo la Teoría de Códigos ha sido fuertemente asociada con los códigos de corrección de errores. Entonces, para fijar ideas, la Teoría de Códigos estudia la comunicación a través de canales ruidosos y trata de cómo asegurar que el mensaje recibido sea el correcto, en contraposición con la criptografía que se ocupa, como ya se expuso, de proteger la comunicación sobre canales no seguros. Es preciso señalar que un sistema de encriptación de diseño de punta hará uso de códigos de corrección de errores al momento de su implementación, porque el cambio en tan sólo 1 bit es suficiente para destruir el mensaje original por completo. Existen dos tipos fundamentales de criptosistemas o sistemas de cifrado: Los criptosistemas simétricos o de clave privada, cuya característica principal es que emplean una misma clave K tanto para cifrar como para descifrar; presentan el inconveniente de que para ser empleados en comunicaciones la clave K debe estar en posesión tanto del emisor como del receptor, lo cual nos lleva a la cuestión de cómo transmitirles a todos los participantes de la comunicación la clave de forma segura. Por otra parte están los criptosistemas asimétricos o de clave pública, que emplean una doble clave (Kp, KP), donde Kp se la conoce como clave privada y KP como clave pública. Una de ellas sirve para la transformación o función E de cifrado y la otra para la transformación D de descifrado. En muchos casos son intercambiables, esto quiere decir que si empleamos una de ellas para cifrar la otra servirá para descifrar y visceversa. Estos criptosistemas deben cumplir además la condición de que el conocimiento de la clave pública KP no permita calcular la clave privada Kp. Tales criptosistemas cumplen a cabalidad el propósito criptográfico de transmitir comunicaciones seguras a través de canales inseguros, puesto que únicamente viaja por el canal la clave pública, que sólo sirve para cifrar o para llevar a cabo autenticaciones. Sin la clave privada (que no es deducible de la clave pública) un observador no autorizado del canal de comunicación será incapaz de descifrar el mensaje cifrado.

En el final de esta introducción cabe mencionar que en Bolivia la investigación en criptología es casi inexistente y por desgracia la legislación boliviana exhibe un gran vacío en estos temas de alta prioridad, así pues todo se reduce a la simple utilización de algoritmos existentes, en cuya implementación surgen los problemas de costos y beneficios, pues no están estandarizados ni adecuados a la administración convencional de datos en general y en todo tipo de instituciones. La criptografía moderna se sustenta en sólidas bases matemáticas, ciencias computacionales y de ingeniería; pero es principalmente el caudal matemático el que provee a esta disciplina de las herramientas para el diseño de los más avanzados sistemas de encriptación y para la generación de sofisticados métodos de ataque, de aquí la importancia de la investigación matemática en este campo. Este trabajo pretende ser una semilla que germine un interés legítimo en la rama de la criptología y por consiguiente tener en el

mediano plazo al menos, profesionales especializados en diseño de criptosistemas, seguridad de la información y hasta manejo del BigData (Procesamiento Masivo de Datos en Bruto) generando oportunidades de incursión laboral en diferentes entidades de diversa índole.

1.2. Introducción

El presente trabajo es una investigación cuyo propósito es el de difundir un campo poco explorado en nuestro ámbito científico como es el de la criptología y más concretamente el estudio de criptosistemas que coadyuven a la seguridad de la información a todo nivel. Se ha estructurado conforme a una secuencia lógica en 14 capítulos.

El capítulo 2 expone los elementos básicos sobre anillos de polinomios y sus propiedades. Se estudia las relaciones de orden en los anillos de polinomios en varias indeterminadas y el algoritmo extendido de la división. El capítulo 3 incluye elementos de campos finitos (donde el campo de Rijndael es un caso particular), construcción de los mismos y representaciones de sus elementos.

El contenido del capítulo 4 atañe en su totalidad a la descripción paso a paso del algoritmo AES y sus especificaciones. Se muestra la programación de subclaves a partir de la clave inicial y se describe los procesos tanto de cifrado como de descifrado. Además se hace el tratamiento de las consideraciones acerca del diseño algorítmico. El capítulo 5 contempla varias de las observaciones hechas al algoritmo y su capacidad de propagar la información, que sugieren la existencia de una fragilidad inherente a la estructura algebraica subyacente.

La parte III y sus capítulos 6, 7 y 8, esta consagrada en su conjunto, al estudio de la propagación y la correlación medidas sobre los datos de entrada o salida y las combinaciones lineales de éstos, todo ésto en el marco de una elegante abstracción algebraica del proceso algorítmico. Asimismo se plantea una interesante relación de identificación entre las propiedades generadas por Funciones Booleanas y las generadas por las funciones lineales en campos finitos.

El capítulo 9 muestra la concreción del estudio hecho en el capítulo anterior acerca de la abstracción del proceso algorítmico denominada como Rijndael-GF, donde a través de la elección de una representación de los elementos del campo finito es obtenido el algoritmo Rijndael como un caso particular. Tales ideas permiten luego la especificación de la función de ronda del cifrado como un polinomio. En el capítulo 10 se lleva a cabo una descripción polinomial del algoritmo Rijndael, basada en un enfoque puramente teórico respecto a las estructuras algebraicas fundamentales que lo cimentan, dejando todo preparado para emprender el cálculo explícito de los coeficientes de la función de ronda echando mano de toda esta maquinaria y herramientas como las bases duales, bases normales y polinomios de permutación. Además se presenta al final la, función de ronda del proceso inverso de desencriptación, con el propósito de

exhibir su complejidad.

Finalmente en el capítulo 11 expongo mis conclusiones y recomendaciones en relación a la consideración y estudio de problemas puntuales que podrían encontrar soluciones en la teoría criptológica.

1.3. Antecedentes

Es el 2 de enero de 1997 cuando el NIST, Instituto Nacional de Normas y Tecnología del gobierno de los Estados Unidos, lanza una convocatoria a un certamen mundial que suscitaría el concurso de una diversidad de propuestas provenientes de la industria y de la comunidad criptográfica, sobre desarrollo en técnicas de encriptación para la protección de datos informáticos de mediana sensibilidad. El 12 de septiembre de ese mismo año se hizo la convocatoria formal. Fue el comienzo de un largo proceso de selección cuyo fin sería la proclamación de un nuevo Estándar de Encriptación Avanzada AES por sus siglas en inglés, que sustituiría al hasta entónces vigente Estándar de Encriptación de Datos DES y de manera gradual al triple-DES; cifrados que en su momento tuvieron un notable éxito siendo adoptados por millones de sistemas en todo el mundo; pero que por sus restricciones de tamaño en bloque y en clave; por su creciente vulnerabilidad y por su bajo rendimiento en relación a los costos de implementación; se encontraron en presencia de una inminente sucesión.

Entre los requerimientos, el nuevo algoritmo debía ser de dominio público y disponible en todo el mundo; aceptar claves de 128, 192 y 256 bits de longitud, operar sobre bloques de entrada de 128 bits además de ofrecer versatilidad y adaptabilidad para trabajar con una amplia variedad de hardware como por ejemplo procesadores de 8 bits que son empleados en tarjetas inteligentes o la muy conocida arquitectura de 32 bits comúnmente usada en computadoras personales, entre otros. La fortaleza criptográfica y la velocidad de procesamiento de datos serían igualmente de alta consideración e importancia en virtud de las demandas de seguridad y optimización de costos.

El 20 de agosto de 1998 el NIST anunció los 15 algoritmos admitidos en la primera conferencia AES. La comunidad criptográfica fue consultada para comentar y evaluar los 15 algoritmos de los cuales cinco finalistas fueron seleccionados: MARS (de IBM), RC6 (de los laboratorios RSA), Rijndael (de los matemáticos belgas Joan Daemen y Vincent Rijmen), Serpent (de Ross Anderson, Eli Biham, y Lars Knudsen),y Twofish (de Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, y Niels Ferguson).

El 15 de mayo del 2000 finalizó el período público de análisis, habiendo ya para entonces el NIST estudiado toda la información disponible para decidir al ganador. En la elección final celebrada el 2 de octubre del 2000, se le confiere al Cifrado por Bloques Rijndael el rótulo de nuevo Estándar de Encriptación Avanzada AES y el 26 de noviembre se publica el FIPS PUB 197 donde se lo asumía oficialmente; un hecho trascendental que marcaría un hito dentro de la criptología y sus avances.

Después de un proceso de estandarización se transformó en un estándar efectivo el 26 de mayo del 2002 y desde entonces se ha convertido en uno de los algoritmos más notables y más populares utilizados en criptografía simétrica. Se trata pues de un diseño de vanguardia elegantemente estructurado a partir la más adecuada y plausible aplicación de la teoría matemática, tanto en la configuración del mismo como en el trazado de estrategias de defensa, desarrolladas para reprimir todos los ataques criptoanalíticos ideados hasta ese momento. Es inevitable formular la siguiente pregunta: ¿Qué pasará con las propuestas que rivalizaron con el algoritmo Rijndael? La respuesta es simple pues dada la jerarquía técnica expuesta por todos y cada uno de los cifrados contendientes, estos se encuentran actualmente a consideración para futuras aplicaciones de diversa índole. Entretanto la investigación se encargará de su constante perfeccionamiento.

1.4. Objetivo del Trabajo

El objetivo central de esta investigación es describir los elementos teórico-matemáticos utilizados en la construcción del algoritmo AES para exhibir claramente las notables ideas propuestas por los diseñadores del algoritmo que resultan en una brillante aplicación práctica que marca además una pauta para la creación de algoritmos o métodos de encriptación.

1.5. Planteamiento del problema

En el presente trabajo se plantea establecer los fundamentos algebraicos, presentes en la Teoría de Números, la Teoría de Campos Finitos y Polinomios, el Álgebra Lineal y la Teoría de Galois, que proporcionan el material necesario para desarrollar el marco teórico en que se sustenta el método de encriptación y descifrado del Estándar de Encriptación Avanzada AES por sus siglas en inglés. Para tal efecto se propone la utilización de campos finitos de característica par en los cuales se definen polinomios de permutación y aplicaciones lineales entre espacios de polinomios que combinados de acuerdo a los lineamientos del diseño, dan estructura a un esquema de cifrado que permitirá la encriptación de datos en general y cuyas especificaciones siguen la directriz del algoritmo de encriptación Rijndael y de una estrategia de diseño conocida como la estrategia del rastro amplio. Se plantea asimismo ilustrar tales herramientas teóricas con ejemplos concretos que dejen visualizar con claridad las aplicaciones del AES.

1.6. Alcance

El alcance del presente trabajo propuesto toma en cuenta los siguientes aspectos:

Alcance Temático: Se sustenta en el plan de estudio de la Carrera de Matemática, en temas como Teoría de Números; Teoría de Campos Finitos; Bases Duales, Normales y de Gröbner; Algebra Lineal; Teoría de la Representación y Probabilidad y Estadística que contextualizan la disciplina de la criptografía y el criptoanálisis, con un enfoque puesto en cifrados por bloques como es el caso del AES.

Alcance Temporal: La información recopilada y analizada corresponde a un lapso temporal que abarca desde el año 1997 hasta el año 2018 y que básicamente cubre desde el proceso de selección del AES, pasando por su estudio en profundidad hasta llegar a las observaciones más trascendentales y su futura sucesión como estándar de encriptación.

1.7. Metodología

El presente trabajo se ha realizado en base a una metodología deductiva, que pone énfasis en la teoría y la abstracción y se apoya en estudios descriptivos enfocados a la divulgación del tema tratado respaldado sólidamente en una cuidadosa exposición de la aplicación de la teoría matemática involucrada.

1.8. Medios

Esta investigación se ha realizado a partir de una extensa revisión bibliográfica que incluye la consulta en la biblioteca especializada de la Carrera de Matemática y la recopilación de información de la red de internet como ser artículos y vasta literatura disponibles todos para el público en general. También se ha empleado material audiovisual cuyo contenido es específico sobre el algoritmo y su funcionamiento.

Parte I

Fundamentos Matemáticos

Capítulo 2

Anillos de Polinomios

Los anillos de polinomios son un ejemplo particular de anillo conmutativo, que desempeña un rol importante en la teoría de campos finitos. El análisis algebraico del AES hace uso extensivo de los anillos de polinomios.

2.1. Anillos de Polinomios Univariantes

Un monomio en una sola variable o indeterminada x es la expresión formal x^i para algún $i \in \mathbb{N}$, ésto es, alguna potencia no negativa de x. El grado del monomio x^i es i.

DEFINICIÓN. Un *polinomio univariante* en la variable x sobre el campo $\mathbb{F}$ es una combinación lineal finita en $\mathbb{F}$ de monomios en x, es decir una expresión formal dada por

$$c_n x^n + c_{n-1} x^{n-1} + \cdots + c_2 x^2 + c_1 x + c_0,$$

donde n es un entero no negativo y $c_n, \ldots, c_0 \in \mathbb{F}$, con $c_n \neq 0$ si $n > 0$.

DEFINICIÓN. El conjunto de todos los polinomios univariantes en la variable x sobre el campo $\mathbb{F}$ dotado con las operaciones usuales de adición y multiplicación de polinomios, forma un anillo. Este anillo es un dominio de ideales principales denominado el *anillo de polinomios univariantes* sobre $\mathbb{F}$ y es denotado por $\mathbb{F}[x]$.

Considere el polinomio univariante $f(x) \in \mathbb{F}[x]$. El grado de $f(x)$ es el máximo entero n tal que $c_n \neq 0$, y es denotado por $grd(f(x))$. Si

$$f(x) = c_n x^n + \cdots + c_1 x + c_0,$$

entonces los sumandos $c_i x^i$ ($c_i \neq 0$) son llamados los términos de $f(x)$, y c_i es llamado el coeficiente del monomio x^i. Asimismo, podemos definir el monomio principal, coeficiente principal y término principal de $f(x)$ como x^n, c_n y $c_n x^n$ respectivamente. Un polinomio $f(x)$ es mónico si su coeficiente principal es 1.

La evaluación del polinomio $f(x)$ en $a \in \mathbb{F}$ es definida como el elemento

$$\sum_{i=0}^{n} c_i a^i \in \mathbb{F}$$

expresado por $f(a)$. Se establece que a es una raíz de $f(x)$ si $f(a) = 0$. Un polinomio de grado n tiene a lo más n raíces en $\mathbb{F}$.

TEOREMA 1. Algoritmo de la División Univariante. Dados $f(x)$ y $g(x)$ en $\mathbb{F}[x]$, entonces existen allí $q(x)$ y $r(x)$ donde o $r(x) = 0$ o bien $grd(r(x)) < grd(g(x))$, tales que $f(x) = q(x)g(x) + r(x)$. El polinomio univariante $r(x)$ es conocido como residuo de la división de $f(x)$ por $g(x)$.

El Algoritmo Euclidiano para encontrar el máximo común divisor de dos polinomios no es más que la aplicación repetida de este teorema.

EJEMPLO. Supongamos que $f(x) = x^6 + x^5 + x^3 + x^2 + x + 1$ y $g(x) = x^4 + x^3 + 1$ son polinomios en el anillo de polinomios univariantes $\mathbb{Z}_2[x]$. Entonces tendremos

$$x^6 + x^5 + x^3 + x^2 + x + 1 = x^2(x^4 + x^3 + 1) + (x^3 + x + 1)$$

así $f(x) = q(x)g(x) + r(x)$ donde $q(x) = x^2$ y $r(x) = x^3 + x + 1$.

Un polinomio $f(x) \in \mathbb{F}[x]$ de grado positivo se dice que es irreducible en $\mathbb{F}[x]$, si no es posible factorizarlo en la forma $f(x) = p(x)q(x)$, donde $p(x)$ y $q(x)$ son polinomios de grado positivo en $\mathbb{F}[x]$. Cada polinomio en $\mathbb{F}[x]$ puede escribirse como producto de polinomios mónicos irreducibles y alguna constante en $\mathbb{F}$ de manera única, salvo el orden de los factores.

EJEMPLO. Sea $f(x)$ un polinomio en $\mathbb{F}[x]$ de grado n, y $\langle f(x) \rangle$ el ideal generado por $f(x)$. Los elementos del anillo cociente $\mathbb{F}[x]/\langle f(x) \rangle$ pueden escribirse como polinomios

$$a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$$

en $\mathbb{F}[x]$ cuyos grados son menores que n. En tal representación del anillo cociente, la adición es simplemente la adición de polinomios. Sin embargo, la multiplicación quedará definida por la aplicación del teorema anterior del modo siguiente. Para dos polinomios $g_1(x), g_2(x) \in \mathbb{F}[x]$, sabemos que existen $q(x), r(x) \in \mathbb{F}[x]$ tales que

$$g_1(x)g_2(x) = q(x)f(x) + r(x),$$

donde $deg(r(x)) < deg(f(x)) = n$. De aquí se sigue que el producto de $g_1(x)$ y$g_2(x)$ es $r(x)$ de acuerdo con la estructura del anillo cociente $\mathbb{F}[x]/\langle f(x) \rangle$.

EJEMPLO. Si $f(x) = x^5 + x^4 + 1$ es un polinomio en el anillo de polinomios univariados $\mathbb{Z}_2[x]$. El producto de los polinomios $(x^4+x^3+x^2+1)$ y (x^4+x^3+x+1) satisface

$$(x^4 + x^3 + x^2 + 1)(x^4 + x^3 + x + 1) = (x^3 + x^2 + x + 1)f(x) + 0.$$

Así en el anillo cociente $R = \mathbb{Z}_2[x]/\langle f(x)\rangle$, el producto de estos dos elementos no nulos es 0, por lo que R no es un dominio de integridad.

TEOREMA 2. El anillo cociente $\mathbb{F}[x]/\langle f(x)\rangle$ es un campo si y sólo si $f(x)$ es irreducible en $\mathbb{F}[x]$.

La Fórmula de Interpolación de Lagrange constituye un método para la construcción de un polinomio a partir de una cantidad finita de valores para evaluación con respecto a una función dada; coincidiendo ambos en tales valores.

TEOREMA 3. Fórmula de Interpolación de Lagrange. Dados $n+1$ pares $(a_i, b_i) \in \mathbb{F} \times \mathbb{F}$, con $a_i \neq a_j$, existe un único polinomio $f(x) \in \mathbb{F}[x]$ de grado a lo más n con $f(a_i) = b_i$. Este polinomio es dado por

$$f(x) = \sum_{i=0}^{n} b_i \prod_{i \neq k=0}^{n} \frac{x - a_k}{a_i - a_k}.$$

2.2. Anillos de Polinomios Multivariantes

Hagamos que $\mathbb{N}^n = \{(\alpha_1, \ldots, \alpha_n) | \alpha_i \in \mathbb{N}\}$ simbolice el conjunto de multi-índices de tamaño n. Un monomio en las variables $x_1, \ldots, x_n$ es un producto de la forma

$$x_1^{\alpha_1} x_2^{\alpha_2}, \ldots, x_n^{\alpha_n},$$

que será expresado por X^α, $\alpha \in \mathbb{N}^n$. El grado de X^α es $d_\alpha = \sum_{i=1}^{n} \alpha_i$.

DEFINICIÓN. Un polinomio en n variables $x_1, \ldots, x_n$ sobre el campo $\mathbb{F}$ es una combinación lineal finita sobre $\mathbb{F}$ de monomios en $x_1, \ldots, x_n$, que constituye una expresión formal dada por

$$\sum_{\alpha \in N} c_\alpha X^\alpha,$$

donde $c_\alpha \in \mathbb{F}$ y N es un subconjunto finito de $\mathbb{N}^n$.

DEFINICIÓN. El conjunto de todos los polinomios en n variables sobre el campo $\mathbb{F}$ forma un anillo bajo las operaciones convencionales de adición y multiplicación de polinomios. Nos referiremos a este anillo como un *anillo de polinomios* sobre $\mathbb{F}$, mismo que para las variables $x_1, \ldots, x_n$ será denotado por $\mathbb{F}[x_1, \ldots, x_n]$.

Sea $f = \sum c_\alpha X^\alpha \in \mathbb{F}[x_1, \ldots, x_n]$ un polinomio multivariante sobre $\mathbb{F}$. Los sumandos $c_\alpha X^\alpha$ ($c_\alpha \neq 0$) son llamados los términos de f y c_α simboliza el coeficiente de X^α. El grado total de f es el máximo entre los grados de todos los monomios de f. Si todos los monomios de f tienen el mismo grado d, diremos que f es homogénea de grado d.

DEFINICIÓN. Si $f \in \mathbb{F}[x_1, \ldots, x_n]$ es un polinomio de grado total d. El polinomio f^h definido por

$$f^h = x_0^d \cdot f\left(\frac{x_1}{x_0}, \ldots, \frac{x_n}{x_0}\right)$$

es un polinomio homogéneo de grado d en $\mathbb{F}[x_0, x_1, \ldots, x_n]$, llamado la homogeneización de f.

Un orden total $\prec$ sobre el conjunto de monomios X^α (donde $\alpha \in \mathbb{N}^n$) que es compatible con la multiplicación es llamado orden monomial en $\mathbb{F}[x_1, \ldots, x_n]$. Un orden es compatible con la multiplicación si $X^\alpha < X^\beta$ implica $X^\alpha X^\gamma < X^\beta X^\gamma$ para todos los multi-índices $\alpha, \beta, \gamma \in \mathbb{N}^n$.

A continuación preséntase tres ejemplos comunes de ordenes monomiales.

DEFINICIÓN. El orden monomial lex (lexicográfico) es definido por $X^\alpha \prec X^\beta$ si la entrada no nula más hacia la izquierda en el vector $\beta - \alpha \in \mathbb{Z}^n$ es positiva.

DEFINICIÓN. El orden monomial glex (grado-lexicográfico) es definido por $X^\alpha \prec X^\beta$ si, en principio $d_\beta > d_\alpha$, y en segunda instancia de ser $d_\beta = d_\alpha$, entonces la entrada no nula más hacia la izquierda en el vector $\beta - \alpha \in \mathbb{Z}^n$ es positiva.

DEFINICIÓN. El orden monomial grevlex (grado-antilexicográfico) es definido por $X^\alpha \prec X^\beta$ si, en principio se tiene que $d_\beta > d_\alpha$, y luego, si $d_\beta = d_\alpha$ entonces la entrada no nula más hacia la derecha en el vector $\beta - \alpha \in \mathbb{Z}^n$ es negativa.

EJEMPLO. Veamos algunos ordenes monomiales en $\mathbb{F}[x, y, z]$.

$$\begin{aligned} \text{ordenamiento lex}: &\quad x^2y^3z^6 < x^2y^4zyxy^3z < x^2yz^2 \\ \text{ordenamiento glex}: &\quad x^2y^4z < x^2y^3z^6yxy^3z < x^2yz^2 \\ \text{ordenamiento grevlex}: &\quad x^2y^4z < x^2y^3z^6yx^2yz^2 < xy^3z \end{aligned}$$

Nótese que el par de monomios x^2y^4z y $x^2y^3z^6$, y el par de monomios xy^3z y x^2yz^2 son ordenados de diferente manera por cada orden monomial.

Supongamos que el anillo de polinomios $\mathbb{F}[x_1, \ldots, x_n]$ tiene un orden monomial $\prec$ y que $f \in \mathbb{F}[x_1, \ldots, x_n]$ es un polinomio. El monomio principal de f es el monomio maximal de f con respecto al orden $\prec$ y es denotado por $LM(f)$. Similarmente, el coeficiente principal de f es el coeficiente del monomio principal de f y es denotado por $LC(f)$. El término principal de f es el término asociado con el monomio principal y es denotado por $LT(f)$, así $LT(f) = LC(f)LM(f)$. El multigrado de f es el grado del monomio principal de f y es denotado por $multigrd(f)$.

Naturalmente, estos conceptos darán lugar a una generalización multivariante del algoritmo de la división univariante.

TEOREMA 4. Algoritmo Polinomial de la División. Supóngase que el anillo polinomial $\mathbb{F}[x_1, \ldots, x_n]$ tiene un orden monomial $\prec$ y que $\{g_1, \ldots, g_s\}$ es un subconjunto ordenado de $\mathbb{F}[x_1, \ldots, x_n]$. Para cualquier $f \in \mathbb{F}[x_1, \ldots, x_n]$, existen $a_i, r \in \mathbb{F}[x_1, \ldots, x_n]$ tales que

$$f = a_1 g_1 + \cdots + a_s g_s + r,$$

donde o bien $r = 0$, o por el contrario $r \neq 0$ y no hay monomio principal alguno de los polinomios g_i que divida a cualquiera de los monomios de r. Tal polinomio r será llamado residuo de la división de f por el conjunto $\{g_1, \ldots, g_s\}$. Además, si $a_i g_i \neq 0$, entonces $multigrd(a_i g_i) \leq multigrd(f)$.

2.3. Álgebras

DEFINICIÓN. Suponga que $\mathcal{A}$ es un espacio vectorial sobre el campo $\mathbb{F}$ con una operación de multiplicación $\mathcal{A} \times \mathcal{A} \to \mathcal{A}$. Si esta operación de multiplicación es asociativa y una aplicación bilineal sobre el espacio vectorial $\mathcal{A}$, entonces $\mathcal{A}$ es un (asociativa) $\mathbb{F}$-*álgebra*, o simplemente un *álgebra*.

Informalmente podemos considerar un álgebra como un espacio vectorial que también es un anillo. La dimensión del álgebra $\mathcal{A}$ es la dimensión de $\mathcal{A}$ como espacio vectorial. El subconjunto $\mathcal{A}' \subset \mathcal{A}$ es un *subálgebra* de $\mathcal{A}$ si $\mathcal{A}'$ es un álgebra *per se* y $\mathcal{A}'$ es un *subálgebra ideal* si es también un ideal del anillo $\mathcal{A}$. Podemos también clasificar las aplicaciones entre dos álgebras de la manera usual, así un homomorfismo de álgebras es una aplicación que es a la vez homomorfismo de anillos y homomorfismo de espacios vectoriales.

EJEMPLO. El anillo de polinomios $\mathbb{F}[x_1, \ldots, x_n]$ es un espacio vectorial sobre $\mathbb{F}$. Así, $\mathbb{F}[x_1, \ldots, x_n]$ forma un $\mathbb{F}$-álgebra, conocida como *álgebra polinomial*.

EJEMPLO. El conjunto $\mathcal{M}_n(\mathbb{F})$ de las matrices $n \times n$ sobre $\mathbb{F}$ forman un espacio vectorial sobre $\mathbb{F}$ de dimensión n^2. La multiplicación de matrices es una aplicación bilineal asociativa sobre $\mathcal{M}_n(\mathbb{F})$. Así $\mathcal{M}_n(\mathbb{F})$ es un $\mathbb{F}$-álgebra de dimensión n^2. El conjunto $\mathcal{D}_n(\mathbb{F})$ de las matrices diagonales $n \times n$ sobre $\mathbb{F}$ forman un subálgebra de $\mathcal{M}_n(\mathbb{F})$ de dimensión n. Tales álgebras son conocidas como *álgebras matriciales*.

Capítulo 3

Propiedades de Campos Finitos

El diseño del AES se basa en el manejo de campos finitos. Todas las operaciones comprendidas en éste algoritmo son descritas por medio de operaciones algebraicas sobre un campo finito de característica par. En esta sección, discutiremos las propiedades más relevantes de los campos finitos para la especificación y el análisis álgebraico del AES.

3.1. Campos Finitos y Subcampos

DEFINICIÓN. Supongamos que $\mathbb{F}$ y $\mathbb{K}$ son dos campos. Si $\mathbb{F} \subset \mathbb{K}$, entonces $\mathbb{F}$ de dice que es un subcampo de $\mathbb{K}$, o de forma equivalente, que $\mathbb{K}$ es un campo de extensión de $\mathbb{F}$.

DEFINICIÓN. Un campo es llamado primo si éste no tiene subcampos propios.

Nótese que cada campo $\mathbb{F}$ contiene un campo primo, la intersección de la familia de sus subcampos y por otra parte que tanto $\mathbb{Q}$ como $\mathbb{Z}_p$ son campos primos. Más aún, éstos son los dos únicos tipos de campos primos.

El conjunto $\mathbb{Z}_p = \{0, \ldots, p-1\}$ con las operaciones de adición y multiplicación en módulo p forma un campo finito si y solo si p es primo. Este es el Campo de Galois de orden p y es denotado por $\mathrm{GF}(p)$. El campo de Galois $\mathrm{GF}(p)$ desempeña un rol fundamental en la teoría de campos finitos.

TEOREMA 1. Un campo finito de característica p (primo) tiene un único subcampo primo isomorfo a $\mathrm{GF}(p)$.

Si $\mathbb{K}$ es un campo de extensión del campo $\mathbb{F}$, entonces $\mathbb{K}$ puede considerarse como un espacio vectorial sobre $\mathbb{F}$. La dimensión de este espacio vectorial es el grado de la extensión. Si $\mathbb{F}$ tiene orden q y $\mathbb{K}$ es un campo extensión de $\mathbb{F}$ de grado d, entonces $\mathbb{K}$ tiene orden q^d. Como cada campo finito tiene característica prima p, se sigue del teorema anterior, que cada campo finito tiene orden p^n para algún primo p y algún entero positivo n.

TEOREMA 2. Para cada número primo p y cada entero positivo n, existe un campo finito de orden p^n. Además, cualesquiera dos campos finitos de orden p^n son isomorfos.

En consecuencia los campos finitos de orden p^n son únicos salvo isomorfismos. Este campo es llamado el Campo de Galois de orden p^n y es denotado por $\mathrm{GF}(p^n)$. Un subcampo de $\mathrm{GF}(p^n)$ tiene orden p^d, donde d es un divisor de n. Además, existe exactamente un subcampo de orden p^d para cada divisor d de n. Por ejemplo, el campo finito $\mathrm{GF}(2^8)$ tiene $\mathrm{GF}(2^4)$, $\mathrm{GF}(2^2)$, y $\mathrm{GF}(2^1)$ como subcampos propios.

TEOREMA 3. El grupo multiplicativo $\mathrm{GF}(q)^* = g \in \mathrm{GF}(q) | g \neq 0$ es un grupo cíclico de orden $q-1$.

Un generador del grupo multiplicativo $\mathrm{GF}(q)^*$ se denomina como elemento primitivo del campo $\mathrm{GF}(q)$. El número de elementos primitivos en $\mathrm{GF}(q)$ es $\phi(q-1)$, donde $\phi(m)$ es el totient de Euler, que da el número de enteros positivos menores que o iguales a m y coprimos a m.

3.2. Construcción Explícita de Campos Finitos

El Teorema 2 del capítulo 2, proporciona un método de construcción como un anillo cociente para un campo finito. Supongamos que $\mathbb{F}$ es un campo finito de orden $q = p^n$ y $f(x) \in \mathbb{F}[x]$ es un polinomio irreducible de grado d. El anillo cociente $\mathbb{K} = \mathbb{F}[x]/\langle f(x) \rangle$ es un campo de orden $q^d = p^{nd}$, siendo éste un campo de extensión de grado d de $\mathbb{F}$. Los elementos de este campo pueden ser representados como

$$a_{d-1}x^{d-1} + \cdots + a_2x^2 + a_1x + a_0,$$

donde $a_i \in \mathbb{F}$. El Teorema 2 del presente capítulo establece que cualquier campo finito de orden p^{nd} es isomorfo a $\mathbb{K}$.

Otra alternativa consiste en construir $\mathrm{GF}(p^{nd})$ como una extensión de campo de $\mathbb{F}$. Sea θ una raíz del polinomio irreducible $f(x)$ de grado d. El conjunto $\mathbb{F}(\theta)$ de todos los cocientes (con denominador no nulo) de polinomios en θ con coeficientes en $\mathbb{F}$ es el más pequeño campo conteniendo a θ y $\mathbb{F}$. Mas precisamente, $\mathbb{F}(\theta)$ es el campo de extensión obtenido incorporando θ a $\mathbb{F}$. $\mathbb{F}(\theta)$ tiene p^{nd} elementos por lo cual es isomorfo a $\mathrm{GF}(p^{nd})$. Los elementos de $\mathbb{F}(\theta)$ están dados por

$$a_{d-1}\theta^{d-1} + \cdots + a_2\theta^2 + a_1\theta + a_0,$$

donde $a_i \in \mathbb{F}$. Si el elemento θ es generador del grupo multiplicativo de $\mathbb{F}(\theta)$, entonces el polinomio $f(x)$ será conocido como polinomio primitivo.

EJEMPLO. El polinomio $m(x) = x^8 + x^4 + x^3 + x + 1 \in \mathrm{GF}(2)[x]$ es irreducible. Si θ es una raíz de $m(x)$, entonces

$$\mathrm{GF}(2)(\theta) \cong \frac{\mathrm{GF}(2)[x]}{\langle m(x) \rangle} \cong \mathrm{GF}(2^8).$$

Los elementos del anillo cociente $\frac{\mathrm{GF}(2)[x]}{\langle m(x)\rangle}$ están dados (para $a_i \in \mathbb{F}$) por

$$a_7x^7 + \cdots + a_2x^2 + a_1x + a_0;$$

mientras que los elementos del campo de extensión $\mathbb{F}(\theta)$ están dados (para $b_i \in \mathbb{F}$) por

$$b_7\theta^7 + \cdots + b_2\theta^2 + b_1\theta + b_0.$$

Nótese que $m(x)$ no es primitivo, puesto que el orden de $\theta \in \mathbb{F}(\theta)$ es 51.

Los polinomios irreducibles sobre un campo $\mathbb{F}$ de orden q son las herramientas básicas para la construcción de todas las extensiones finitas de $\mathbb{F}$. Si $\mathbb{K}$ es una extensión de $\mathbb{F}$ de orden q^n, entonces el Teorema 3 de este capítulo, muestra que $a^{q^n-1}-1 = 0$ para todo $a \in \mathbb{K}$ no nulo. Luego, el polinomio $x^{q^n}-x$ tiene a cada uno de los q^n elementos de $\mathbb{K}$ como una raíz. El campo $\mathbb{K} \cong \mathrm{GF}(q^n)$ es conocido como el campo de descomposición del polinomio $x^{q^n} - x$. Este polinomio puede ser usado para obtener todos los polinomios irreducibles sobre $\mathbb{F}$ con el grado requerido.

TEOREMA 4. Si $\mathbb{F}$ es un campo finito de orden q. entonces el polinomio $x^{q^n} - x \in \mathbb{F}[x]$ es el producto de todos los polinomios mónicos irreducibles en $\mathbb{F}[x]$ cuyos grados dividen a n.

El número de polinomios irreducibles en $\mathbb{F}[x]$ de grado n es dado por

$$\frac{1}{n}\sum_{d|n}\mu(d)q^{\frac{n}{d}},$$

donde μ es la función de Möbius, definida por $\mu(1) = 1$, $\mu(n) = (-1)^k$ si n es el producto de k distintos primos, y 0 en cualquier otro caso. El número de polinomios primitivos de grado n es $\frac{1}{n}\varphi(q^n - 1)$, donde φ es el totient de Euler.

EJEMPLO. Existen $\frac{1}{8}(\mu(1)2^8+\mu(2)2^4+\mu(4)2^2+\mu(8)2^1) = 60$ polinomios irreducibles de grado 8 en $\mathrm{GF}(2)[x]$, de los que $\frac{1}{8}\varphi(2^8 - 1) = 16$ son polinomios primitivos.

DEFINICIÓN. Un campo $\mathbb{F}$ se dice que es algebraicamente cerrado si cada polinomio en $\mathbb{F}[x]$ tiene una raíz en $\mathbb{F}$. Además la clausura algebraica de un campo $\mathbb{F}$ es el más pequeño campo extensión $\mathbb{K}$ de $\mathbb{F}$ tal que $\mathbb{K}$ sea algebraicamente cerrado.

3.3. Representaciones para un Campo Finito

Sea $\mathbb{F}$ un campo y $\mathbb{K} = \mathbb{F}(\theta)$ una extensión de campo de $\mathbb{F}$ de grado d. La manera más natural de describir los elementos de $\mathbb{K}$ es considerarlos como vectores del espacio vectorial $\mathbb{K}$ de dimensión d sobre $\mathbb{F}$. Cada elemento en $\mathbb{K}$ puede escribirse de manera unívoca como

$$a_{d-1}\theta^{d-1} + \cdots + a_2\theta^2 + a_1\theta + a_0,$$

donde $a_i \in \mathbb{F}$. Así, el conjunto $\{\theta^{d-1}, \ldots, \theta^2, \theta, 1\}$ conforma una base del espacio vectorial d-dimensional $\mathbb{K}$ sobre $\mathbb{F}$. Esta base se denomina base polinomial para el campo $\mathbb{K}$.

EJEMPLO. Suponga que θ es una raíz de $x^8 + x^4 + x^3 + x + 1 \in \mathrm{GF}(2)[x]$, y que $\mathbb{K}$ es el campo $\mathrm{GF}(2)(\theta)$. Cualquier aplicación multiplicación $\mathbb{K} \to \mathbb{K}$ es una transformación lineal de $\mathbb{K}$ como espacio vectorial sobre $\mathrm{GF}(2)$. La aplicación al-cuadrado en $\mathbb{K}$ es también una transformación lineal. Si T_θ y S denotan las matrices que corresponden a la multiplicación por θ y a al-cuadrado con respecto a la base polinomial $\{\theta^7, \ldots, \theta^2, \theta, 1\}$, entonces tenemos

$$T_\theta = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, S = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Existen otras bases que pueden ser usadas por el campo $\mathbb{K}$ cuando es considerado como un espacio vectorial sobre $\mathbb{F}$, tales como las bases normales $\{\beta, \beta^q, \beta^{q^2}, \ldots, \beta^{q^{d-1}}\}$ para un $\beta \in \mathbb{K}$ adecuado. Esta representación es particularmente útil cuando se requiere manipular la exponenciación de los elementos de $\mathbb{K}$, además de ofrecer en ciertas situaciones ventajas en la implementación.

Se cuentan también con métodos para describir los elementos del campo finito $\mathbb{K}$ de orden q^n que dependen de funciones logarítmicas definidas en $\mathbb{K}$; mas no de aspectos relativos a su carácter de espacio vectorial. Supóngase que β es un elemento primitivo de $\mathbb{K}$ y $a = \beta^i$ ($0 \leq i < q^n - 1$). El logaritmo discreto es la función $\log_\beta : \mathbb{K}^* \to \mathbb{Z}_{q^n-1}$ definida por $\log_\beta a = \log_\beta \beta^i = i$. Así, los elementos no nulos $a \in \mathbb{K}$ pueden ser representados por $\log_\beta a \in \mathbb{Z}_{q^n-1}$. Adoptando la convención de que el logaritmo discreto de 0 es denotado por ∞, entonces cada elemento de $\mathbb{K}$ queda identificado con un elemento de $\overline{\mathbb{Z}}_{q^n-1} = \mathbb{Z}_{q^n-1} \cup \{\infty\}$.

El logaritmo de Zech o de Jacobi ofrece otro método logarítmico para describir los elementos de un campo finito. El logaritmo de Zech se basa en la función $Z : \overline{\mathbb{Z}}_{q^n-1} \to \overline{\mathbb{Z}}_{q^n-1}$ dada por

$$Z(n) = \log_\beta(\beta^n + 1),$$

así $\beta^{Z(n)} = \beta^n + 1$ con la convención $\beta^\infty = 0$. La definición puede ser extendida a todos los enteros trabajando en módulo $q^n - 1$. El logaritmo de Zech de β^n ahora puede ser definido por $Z(n)$. Tenemos las siguientes identidades concernientes a la función Z:

$$\begin{aligned} Z(Z(n)) &= n, \\ Z(2n) &= 2Z(n), \\ Z(-n) &= Z(n) - n. \end{aligned}$$

Esta función es de interés pues puede usarse para calcular la suma de dos potencias de β,

$$\beta^m + \beta^n = \beta^n(\beta^{m-n} + 1) = \beta^n \beta^{Z(m-n)} = \beta^{n+Z(m-n)}.$$

3.4. Funciones en un Campo Finito

DEFINICIÓN. Sea $\mathbb{F}$ un campo finito de orden q y $\mathbb{K}$ una extensión de campo de $\mathbb{F}$ de grado d. Los elementos $a, a^q, a^{q^2}, \ldots, a^{q^{d-1}}$ son los *conjugados* de $a \in \mathbb{K}$ con respecto de $\mathbb{F}$.

TEOREMA 5. Supongamons que $\mathbb{K}$ es una extensión de campo de grado d de un campo $\mathbb{F}$. Cualquier elemento $a \in \mathbb{K}$ es una raíz de un polinomio irreducible $f(x) \in \mathbb{F}[x]$ de grado n divisor de d. Las raíces de $f(x)$ son los conjugados de a.

A continuación ponemos en consideración algunas funciones de interés en campos finitos.

DEFINICIÓN. Sea $\mathbb{F}$ un campo finito de orden q y $\mathbb{K}$ un campo extensión de grado d para $\mathbb{F}$. La función *traza* sobre $\mathbb{K}$ con respecto a $\mathbb{F}$ es la función $\mathrm{Tr}_{\mathbb{K}/\mathbb{F}} : \mathbb{K} \to \mathbb{F}$ definida por

$$\mathrm{Tr}_{\mathbb{K}/\mathbb{F}}(a) = a + a^q + a^{q^2} + \cdots + a^{q^{d-1}}.$$

Luego la traza de un elemento $a \in \mathbb{K}$ es la suma de todos los conjugados de a. La función traza es un funcional lineal sobre $\mathbb{K}$, considerado como un espacio vectorial sobre $\mathbb{F}$. De hecho, cualquier funcional lineal sobre $\mathbb{K}$ es de la forma $a \mapsto \mathrm{Tr}_{\mathbb{K}/\mathbb{F}}(\beta a)$ para algún $\beta \in \mathbb{K}$.

DEFINICIÓN. Sea $\mathbb{F}$ un campo finito de orden q y $\mathbb{K}$ una campo extensión de grado d para $\mathbb{F}$. La función *norma* sobre $\mathbb{K}$ con respecto a $\mathbb{F}$ es la función $\mathrm{N}_{\mathbb{K}/\mathbb{F}} : \mathbb{K} \to \mathbb{F}$ definida por

$$\mathrm{N}_{\mathbb{K}/\mathbb{F}}(a) = a a^q a^{q^2} \cdots a^{q^{d-1}} = a^{\frac{q^d-1}{q-1}}.$$

Luego la norma de un elemento $a \in \mathbb{K}$ es el producto de todos los conjugados de a. La función norma es un homomorfismo de grupos $\mathbb{K}^* \to \mathbb{F}^*$ entre los grupos multiplicativos de los campos $\mathbb{K}$ y $\mathbb{F}$.

DEFINICIÓN. Un *polinomio linealizado* $f(x) \in \mathbb{K}[x]$ es un polinomio dado por

$$f(x) = a_0 x + a_1 x^q + a_2 x^{q^2} + \cdots + a_{d-1} x^{q^{d-1}},$$

donde $a_i \in \mathbb{K}$. Entonces un polinomio linealizado $f(x)$ es un polinomio cuya evaluación $f(a)$ para cada $a \in \mathbb{K}$ resulta en una combinación lineal de los d conjugados de a.

Los polinomios linealizados son transformaciones lineales sobre $\mathbb{K}$ considerado como un espacio vectorial sobre $\mathbb{F}$. De manera recíproca, cualquier transformación lineal de $\mathbb{K}$ sobre $\mathbb{F}$ puede expresarse como un polinomio linealizado, como se ve en el próximo ejemplo.

EJEMPLO. Dado que cualquier transformación lineal de $\mathrm{GF}(2^8)$ considerado como un espacio vectorial sobre $\mathrm{GF}(2)$, puede ser representada por un polinomio (linealizado) de la forma $f(x) = a_0x^{2^0} + a_1x^{2^1} + a_2x^{2^2} + \cdots + a_7x^{2^7}$, donde $a_i \in \mathrm{GF}(2^8)$.

La prueba de este hecho se sigue del Teorema 6 al final de esta sección; pero previamente tomemos en cuenta ciertas consideraciones. Sea el campo $\mathrm{GF}(p^d)$ una extensión de campo de $\mathrm{GF}(p)$, donde p es primo. La función $\sigma : \mathrm{GF}(p^d) \to \mathrm{GF}(p^d)$ definida por $a \mapsto a^p$ aplica a a uno de sus conjugados con respecto a $\mathrm{GF}(p^d)$. Esta aplicación satisface

$$\sigma(a + a') = \sigma(a) + \sigma(a') \text{ y } \sigma(aa') = \sigma(a)\sigma(a').$$

Tenemos que σ es un automorfismo de campos de $\mathrm{GF}(p^d)$, conocido como el automorfismo de *Frobenius*. El conjunto de todos los automorfismos de $\mathrm{GF}(p^d)$ es el grupo cíclico de orden d generado por σ bajo la operación de composición. Nótese que σ mantiene fijos todos los elementos del subcampo $\mathrm{GF}(p)$ de $\mathrm{GF}(p^d)$. Así, los automorfismos de $\mathrm{GF}(p^d)$ son también transformaciones lineales sobre $\mathrm{GF}(p)$.

Tenemos a continuación la siguiente caracterización del álgebra $End(\mathbb{F}_{q^n})$ de de las transformaciones lineales de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$.

Sea $\mathcal{L}_n(\mathbb{F}_{q^n})$ el conjunto de todos los polinomios linealizados, claramente un álgebra sobre $\mathbb{F}_{q^n}$. Se sabe que $End(\mathbb{F}_{q^n})$ es un espacio vectorial sobre $\mathbb{F}_q$; pero puede convertirse en un espacio vectorial sobre $\mathbb{F}_{q^n}$, introduciendo la multiplicación escalar

$$(aT)(x) = aT(x), x \in \mathbb{F}_{q^n},$$

Cualesquiera sean $a \in \mathbb{F}_{q^n}$ y $T \in End(\mathbb{F}_{q^n})$. Nótese que $\{\sigma^i \in End(\mathbb{F}_{q^n}) | 0 \leq i \leq n-1\}$ es un conjunto linealmente independiente sobre $\mathbb{F}_{q^n}$, porque de lo contrario el polinomio $\sum_{i=0}^{n-1} a_i\sigma^i(x) = \sum_{i=0}^{n-1} a_i x^{q^i}$ admitirá q^n raíces para ciertos $a_0, \ldots, a_{n-1} \in \mathbb{F}_{q^n}$ de los cuales no todos son nulos. Dado que $dim_{\mathbb{F}_q} End(\mathbb{F}_{q^n}) = n^2$ y $dim_{\mathbb{F}_q}\mathbb{F}_{q^n} = n$, se tiene que

$$dim_{\mathbb{F}^{q^n}} End(\mathbb{F}_{q^n}) = n,$$

lo cual implica que para cualquier $T \in End(\mathbb{F}_{q^n})$, existen $a_0, \ldots, a_{n-1} \in \mathbb{F}_{q^n}$ tales que $T = \sum_{i=0}^{n-1} a_i\sigma^i$.

Defínase la función

$$End(\mathbb{F}_{q^n}) \to \mathcal{L}_n(\mathbb{F}_{q^n}), \quad \sum_{i=0}^{n-1} a_i\sigma^i \mapsto \sum_{i=0}^{n-1} a_i x^{q^i}$$

Es fácil ver que esta función es biyectiva. Más aún, de manera directa se puede verificar que la función arriba definida es un isomorfismo de $\mathbb{F}_q$-álgebras.

Queda entonces demostrado el siguiente resultado que nos permitirá de ahora en máa, manejar indistintamente aplicaciones lineales y los polinomios linealizados en $\mathbb{F}_{q^n}$.

TEOREMA 6. $\mathcal{L}_n(\mathbb{F}_{q^n}) \cong End(\mathbb{F}_{q^n})$.

3.5. Funciones Booleanas

Primeramente, daremos la definicion de un vector Booleano, el cual es como veremos, la entrada y la salida del algoritmo Rijndael.

DEFINICIÓN. Un vector Booleano b de longitud n es un vector cuyas entradas son bits:

$$b \in \mathrm{GF}(2)^n.$$

Un *vector Booleano* de longitud n es llamado también un arreglo unidimensional de bits de longitud n.

Rijndael es un cifrado que opera sobre bytes, mismos que pueden identificarse con elementos del campo finito $\mathrm{GF}(2^8)$. Por otra parte cualquier vector Booleano de longitud n representa un elemento del campo finito $\mathrm{GF}(2^n)$. Ahora definiremos una función Booleana que opera sobre el campo finito $\mathrm{GF}(2^n)$ visto como un espacio vectorial sobre el campo $\mathrm{GF}(2)$.

DEFINICIÓN. Una *función Booleana* ϕ, es una aplicación que le hace corresponder a un vector Booleano otro vector Booleano:

$$\phi : \mathrm{GF}(2)^n \to \mathrm{GF}(2)^m.$$

Una *transformación Booleana* χ, es una función Booleana que aplica un vector Booleano en otro vector Booleano de la misma longitud:

$$\chi : \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n.$$

Diremos en este caso que opera sobre un un estado de n bits. Si la transformación Booleana χ es invetible, ésta es llamada una *permutación Booleana en serie*.

Vamos ahora con tres funciones Booleanas especiales. La función bricklayer, la transposición y la función Booleana iterativa. Como veremos Rijndael es una permutación Booleana iterativa, la cual concatena muchas funciones de ronda Booleanas. Éstas son a su vez también permutaciones Booleanas iterativas, las cuales consisten de tres permutaciones Booleanas individuales, denominadas permutación bricklayer no lineal, transposición y permutación bricklayer lineal.

DEFINICIÓN. Una *función bricklayer* es una función Booleana que puede ser descompuesta en varias funciones Booleanas, operando independientemente sobre conjuntos de bits del vector entrante. Si éstas funciones Booleanas son lineales, son llamadas *cajas de difusión* o *D-box* y si son no lineales, son llamadas *cajas de sustitución* o *S-box*.
Una *transformación (permutación) bricklayer* es una función bricklayer, la cual puede descomponerse en varias transformaciones (permutaciones) Booleanas.
Una transposición es una permutación Booleana, para la cual el vector binario saliente tiene el mismo peso de Hamming (número de de entradas no nulas en el vector) que el vector entrante.

DEFINICIÓN. Una *función Booleana iterativa* $\psi : \mathrm{GF}(2^{n_0}) \to \mathrm{GF}(2)^{n_k}$ es la concatenación de $k \in \mathbb{N}$ funciones Booleanas $\psi_i : \mathrm{GF}(2)^{n_i} \to \mathrm{GF}(2)^{n_{i+1}}$, para $i \in \{0, \ldots, k-1\}$:

$$\psi = \psi_{k-1} \circ \cdots \circ \psi_0.$$

Una *transformación (permutación) Booleana iterativa* es la concatenación de transformaciones (permutaciones) Booleanas.

Parte II

Descripción del Algoritmo AES y sus Especificaciones

Capítulo 4

Estándar de Encriptación Avanzada AES

Rijndael es un cifrado iterado por bloques. Ésto significa que una sucesión de cómputos denominada ronda, es repetida un determinado número de veces. Su diseño permite el manejo de longitudes variables tanto en bloque de estado como en clave. Ambas longitudes pueden especificarse independientemente como cualquier múltiplo de 32 bits, con un mínimo de 128 y un máximo de 256 bits. Este estándar especifica el algoritmo Rijndael como un cifrado por bloques simétrico, capaz de procesar bloques de datos de 128 bits y utiliza claves de longitudes 128, 192, y 256 bits, correspondiendo a cada caso las denominaciones de «AES-128», «AES-192» y «AES-256». Los tamaños adicionales en bloque, consignados en la propuesta original del algoritmo Rijndael, no son adoptados por este estándar. Así pues, se seguirá la descripción formal del algoritmo dada en el documento FIPS 197, anuncio del AES.

El siguiente es un breve bosquejo del algoritmo AES. Más adelante se expondrá más detalle con respecto a su especificación.

El algoritmo AES consiste de una ronda inicial y otras 10, 12 o 14 rondas, dependiendo del tamaño de la clave. Cada ronda incluye una clave de ronda, que se deriva de la clave principal. Cada ronda comienza con un bloque de entrada de 128 bits que produce un bloque de salida de 128 bits.

Se emplean cuatro etapas básicos, mismas que constituyen una ronda:

1. **`SubBytes`**: Esta etapa no lineal causa confusión, con el objetivo de tornar compleja la relación entre el texto-franco, el texto-cifrado y la clave, proporcionando resistencia contra los ataques criptoanalíticos lineal y diferencial.

2. **`ShiftRows`**: Es una transformación lineal y causa difusión, con el propósito de extender la influencia de todas los datos de entrada y la clave sobre todos los datos de salida.

3. **`MixColumns`**: Esta etapa tiene un propósito similar a la anterior proporcionando dispersión a nivel de los bytes.

4. **`AddRoundKey`**: Aquí la clave de ronda es añadida por medio de la operación XOR con el resultado obtenido en la etapa anterior.

Entonces la estructura de una ronda es:

$$\longrightarrow \boxed{\texttt{SubBytes}} \longrightarrow \boxed{\texttt{ShiftRows}} \longrightarrow \boxed{\texttt{MixColumns}} \longrightarrow \boxed{\texttt{AddRoundKey}} \longrightarrow$$

Concatenando todo, con la abreviación correspondiente, obtenemos lo siguiente:

Encriptación vía Rijndael	
Ronda	Vinculación de Etapas
0	ARK, usando la clave inicial.
1	SB, SR, MC, ARK, usando la clave 1.
2	SB, SR, MC, ARK, usando la clave 2.
⋮	⋮
9	SB, SR, MC, ARK, usando la clave 9.
10	SB, SR, ARK, usando la clave 10.

Los 128 bits de salida constituyen el bloque cifrado.

4.1. Notación

La notación presentada a continuación corresponde al contenido del anuncio oficial del algoritmo Rijndael como el AES. No obstante es preciso señalar que su uso esta restringido mayormente al presente capítulo, en virtud de proporcionar un lógico desarrollo de los tópicos tratados y en virtud también del marco de referencia considerado.

4.1.1. Entradas y Salidas

Tanto una entrada como una salida para el AES, consisten de secuencias de 128 *bits* (dígitos con valores de 0 y 1). Nos referiremos a tales secuencias a veces como bloques de tamaño 128. La clave del cifrado para el AES es una secuencia de 128, 192 o 256 bits. Tamaños diferentes, no están permitidos en este estándar.

Los bits de las secuencias, serán enumerados comenzando de cero y terminando en el valor del tamaño de la secuencia (tamaño en bloque o longitud de clave) menos 1. El número i asignado a un bit será el índice y variará en los siguientes rangos $0 \leq i < 128$, $0 \leq i < 192$ o $0 \leq i < 256$ dependiendo del caso.

4.1.2. Bytes

La unidad básica de procesamiento en el AES es un *byte*, una secuencia de 8 bits considerada como una entidad individual. La entrada, la salida y la clave son procesadas como arreglos de bytes, arreglos generados dividiendo estas secuencias en grupos de 8 bits. Para una entrada, salida o clave denotada por a, los bytes en el arreglo resultante serán referenciados como a_n, donde n estará en uno de los siguientes rangos:

$$\begin{aligned}\text{Tamaño en bloque} &= 128bits, 0 \leq n < 16;\\ \text{Longitud de clave} &= 128bits, 0 \leq n < 16;\\ \text{Longitud de clave} &= 128bits, 0 \leq n < 16;\\ \text{Longitud de clave} &= 128bits, 0 \leq n < 16.\end{aligned}$$

Todos los valores de los bytes en el AES, serán presentados como la concatenación de sus valores individuales de bits (0 o 1) en la forma $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$. Estos bytes son interpretados como elementos de un campo finito, empleando representación polinomial:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_ix^i.$$

Por ejemplo, {`01100011`} identifica al elemento $x^6 + x^5 + x + 1$ del campo finito.

Es conveniente denotar los valores de los bytes mediante el uso de la notación hexadecimal, con cada patrón de cuatro bits representado por un caracter como se ve a continuación:

Binario	Hexadecimal
`0000`	**0**
`0001`	**1**
`0010`	**2**
`0011`	**3**

Binario	Hexadecimal
`0100`	**4**
`0101`	**5**
`0110`	**6**
`0111`	**7**

Binario	Hexadecimal
`1000`	**8**
`1001`	**9**
`1010`	**A**
`1011`	**B**

Binario	Hexadecimal
`1100`	**C**
`1101`	**D**
`1110`	**E**
`1111`	**F**

Figura 1. Representación hexadecimal de patrones binarios.

De aquí que el elemento {`01100011`} puede ser representado como {**63**}

Los arreglos de bytes serán representados en la forma siguiente:

$$a_0a_1a_2 \ldots a_{15}$$

Los propios bytes y el ordenamiento de sus bits componentes, se derivan de la secuencia de entradas de 128 bits

$$entrada_0 entradda_1 entrada_2 \ldots entrada_{126} entrada_{127}$$

de acuerdo al esquema general

$$a_n = \{entrada_{8n}, entrada_{8n+1}, \ldots, entrada_{8n+7}\}$$

donde $0 \leq n < 16$, $0 \leq n < 24$ o $0 \leq n < 32$, dependiendo del tamaño de la clave.

4.1.3. El Estado

Internamente, las operaciones del algoritmo AES son aplicadas sobre un arreglo 2-dimensional de bytes llamado *el estado* o *el arreglo de estados*. El estado consiste de 4 filas dde bytes, cada uno conteniendo Nb bytes, donde Nb es el tamaño en bloque dividido por 32. Cada byte en el arrreglo del estado, será representado por $s_{i,j}$ donde $0 \leq i < 4$ y $0 \leq j < Nb$. Para este estándar, $Nb = 4$.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

Figura 2. El arreglo de estados.

Dada la secuencia de bytes $a_0, a_1, \ldots, a_{15}$ al principio de los procesos de encriptación y desencriptación representando un texto-franco o un texto-cifrado respectivamente, ésta será copiada en el estado de acuedo a la relación

$$s_{i,j} = a_{4j+i}, \quad \text{para } 0 \leq i < 4 \text{ y } 0 \leq j < Nb$$

Además, obtenido un arreglo de estados al final de los procesos de encriptación y desencriptación, éste será dispuesto en una secuencia de bytes $a'_0, a'_1, \ldots, a'_{15}$ representando un texto-cifrado o un texto-franco respectivamente de acuerdo a la relación

$$a'_{4j+i} = s_{i,j}, \quad \text{para } 0 \leq i < 4 \text{ y } 0 \leq j < Nb$$

Los cuatro bytes en cada columna del arreglo de estados forman palabras de 32 bits, donde el número de fila r proporciona un índice para los cuatro bytes dentro de cada palabra. El estado puede verse entonces como un arreglo unidimensional de palabras (columnas) de 32 bits, $w_0 w_1 w_2 w_3$, donde la columna número c provee un índice para este arreglo. De aquí que el estado puede verse como un arreglo de cuatro palabras, como sigue:

$$w_m = [s_{0,m} s_{1,m} s_{2,m} s_{3,m}], \quad \text{donde } 0 \leq m < 4$$

4.2. El Sustento Matemático

Un byte es identificado con un elemento del campo finito $\mathrm{GF}(2^8)$. El estándar AES especifica una representación para un byte en $\mathrm{GF}(2^8)$ a través de la construcción de este campo en términos del polinomio

$$m(x) = x^8 + x^4 + x^3 + x + 1,$$

que es irreducible en $\mathrm{GF}(2)[x]$ (ver ejemplo en la sección 3.2). Llamaremos a éste polinomio como *el Polinomio de Rijndael*. Como ya vimos, el campo $\mathrm{GF}(2^8)$ puede construirse o bien como anillo cociente, o como una extensión de campo con respecto al polinomio señalado. Nos referiremos al campo $\mathrm{GF}(2^8)$ definido por el polinomio de Rijndael como *el Campo de Rijndael* y será denotado, aparte de la forma convencional, como $\mathbb{F}$. Entonces $\mathbb{F} = \frac{\mathrm{GF}(2)[x]}{\langle m(x)\rangle}$ o $\mathbb{F} = \mathrm{GF}(2)(\theta)$, donde θ denota una raíz del polinomio de Rijndael, llamada *la Raíz Rijndael*. En síntesis, la representación de un byte $\{b_7b_6b_5b_4b_3b_2b_1b_0\}$ en $\mathbb{F}$ puede ser dada en cualquiera de las siguientes formas equivalentes:

$$\begin{aligned} \mathbb{F} \text{ como Anillo Cociente} &: b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 \\ \mathbb{F} \text{ como Extesión de Campo} &: b_7\theta^7 + b_6\theta^6 + b_5\theta^5 + b_4\theta^4 + b_3\theta^3 + b_2\theta^2 + b_1\theta + b_0 \end{aligned}$$

En el AES los bytes así representados se combinan con las operaciones de adición y multiplicación en el campo, considerando que la adición es la operación Booleana XOR que expresaremos mediante el símbolo $\oplus$. Asimismo la multiplicación estará denotada por el símbolo $\bullet$.

Multiplicar un polinomio binario de $\mathbb{F}$ por el polinomio x da como resultado

$$b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x.$$

Para obtener el resultado de este producto se debe hacer la reducción módulo $m(x)$. Si $b_7 = 0$, el resultado está ya en la forma reducida. Si $b_7 = 1$, la reducción se realiza adicionándole el polinomio $m(x)$. Se sigue que la multiplicación por x ($\{$`00000010`$\}$ o $\{$**`02`**$\}$) puede ser implementada a nivel de bytes desplazando un bit a la izquierda y realizando la operación XOR con $\{$**`1B`**$\}$. Esta operación se denota por `xtime`. La multiplicación por potencias superiores de x puede implementarse a través de aplicaciones repetidas de `xtime` y puede implementarse también el producto por cualquier constante mediante la adición de resultados intermedios.

4.2.1. Polinomios con Coeficientes en $\mathrm{GF}(2^8)$

Polinomios de cuatro términos con coeficientes que están en el campo finito $\mathbb{F}$, se pueden definir como:

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

el cual será denotado como una palabra de la forma $[a_0, a_1, a_2, a_3]$. Nótese que tales polinomios se comportan de modo un tanto diferente a los polinomios que denotan elementos de un campo finito. Los coeficientes son éllos mismos elementos del campo finito $\mathbb{F}$, dicho de otra manera, los coeficientes son ahora bytes en vez de bits. Además la multiplicación utiliza una reducción polinomial diferente definida más abajo. La distinción de indeterminadas y coeficientes queda clara desde el contexto.

Para ilustrar las operaciones de adición y multiplicación, sea

$$b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$$

un segundo polinomio de cuatro términos. La adición es llevada a cabo sumando coeficientes de potencias similares. Dado que los coeficientes son elementos del campo $\mathbb{F}$ la suma de los mismos corresponde a una operación XOR entre bytes. Luego tenemos

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0)$$

donde $\oplus$ recordemos, indica la adición en $\mathbb{F}$.

La multiplicación es llevada a cabo en dos pasos. En el primero, el polinomio producto $c(x) = a(x) \cdot b(x)$, es expandido reuniendo potencias similares, dando como resultado

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$

donde

$$\begin{aligned}
c_0 &= a_0 \bullet b_0 \\
c_1 &= a_1 \bullet b_0 \oplus a_0 \bullet b_1 \\
c_2 &= a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2 \\
c_3 &= a_3 \bullet b_0 \oplus a_2 \bullet b_1 \oplus a_1 \bullet b_2 \oplus a_0 \bullet b_3 \\
c_4 &= a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet b_3 \\
c_5 &= a_3 \bullet b_2 \oplus a_2 \bullet b_3 \\
c_6 &= a_3 \bullet b_3
\end{aligned}$$

Este polinomio no representa una palabra de cuatro bytes. Por consiguiente, el segundo paso de la multiplicación será llevar a cabo la reducción de $c(x)$ módulo un polinomio de grado 4; el resultado será un polinomio de grado menor a 4. Para el algoritmo AES, ésto se efectúa con la utilización del polinomio $x^4 + 1$, tal que

$$x^i \mod (x^4 + 1) = x^{i \mod 4}.$$

El producto modular de $a(x)$ y $b(x)$, denotado por $a(x) \otimes b(x)$, está dado por el polinomio de cuatro términos $d(x)$, definido a continuación:

$$d(x) = d_3x^3 + d_2x^2 + d_1x + d_0$$

con

$$\begin{aligned}
d_0 &= (a_0 \bullet b_0) \oplus (a_3 \bullet b_1) \oplus (a_2 \bullet b_2) \oplus (a_1 \bullet b_3) \\
d_1 &= (a_1 \bullet b_0) \oplus (a_0 \bullet b_1) \oplus (a_3 \bullet b_2) \oplus (a_2 \bullet b_3) \\
d_2 &= (a_2 \bullet b_0) \oplus (a_1 \bullet b_1) \oplus (a_0 \bullet b_2) \oplus (a_3 \bullet b_3) \\
d_3 &= (a_3 \bullet b_0) \oplus (a_2 \bullet b_1) \oplus (a_1 \bullet b_2) \oplus (a_0 \bullet b_3)
\end{aligned}$$

Cuando $a(x)$ es un polinomio fijo, esta operación puede ser escrita en forma matricial como:

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

Por el hecho de que $x^4 + 1$ no es un polinomio irreducible sobre GF(2^8), la multiplicación por un polinomio fijo de cuatro términos no necesariamente es invertible. Sin embargo, el algoritmo AES especifica un polinomio de cuatro términos que si tiene inverso.

$$\begin{aligned}
a(x) &= \{\mathbf{03}\}x^3 + \{\mathbf{01}\}x^2 + \{\mathbf{01}\}x + \{\mathbf{02}\} \\
a^{-1}(x) &= \{\mathbf{0B}\}x^3 + \{\mathbf{0D}\}x^2 + \{\mathbf{09}\}x + \{\mathbf{0E}\}
\end{aligned}$$

Otro polinomio utilizado en el AES, más concretamente en el proceso de expansión de claves, tiene $a_0 = a_1 = a_2 = \{\mathbf{00}\}$ y $a_3 = \{\mathbf{01}\}$, el cual es el polinomio x^3. Su efecto es formar una palabra saliente por acción de una rotación de los bytes en la palabra entrante. Esto significa que $[b_0, b_1, b_2, b_3]$ es transformada en $[b_1, b_2, b_3, b_0]$.

4.3. Especificaciones del Algoritmo

Formalmente tenemos que para el algoritmo AES, el tamaño en bloque de entrada, en bloque de salida y en el estado es de 128 bits . Esto está representado por $Nb = 4$, lo cual refleja el número de palabras de 32 bits (número de columnas) en el estado.

Para el algoritmo AES, la longitud de clave del cifrado K, es 128, 192 o 256 bits. La longitud de la clave está representada por $Nk = 4, 6$ u 8, lo cual refleja el número de palabras de 32 bits (número de columnas) en la clave del cifrado.

Para el algoritmo AES, el número de rondas a ser ejecutadas mientras corre el algoritmo, depende de la longitud de clave. El número de rondas está representado por Nr, donde $Nr = 10$ cuando $Nk = 4$, $Nr = 12$ cuando $Nk = 6$ y $Nr = 14$ cuando $Nk = 8$.

	Nk	Nb	Nr
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Figura 3. Combinaciones de clave-bloque-ronda.

Las únicas combinaciones de clave-bloque-ronda que se contemplan en este estándar estan dadas en la Figura 3. Sin embargo, futuras reafirmaciones de este estándar podrían incluir cambios o ampliaciones a la lista de valores permitidos para aquéllos parámetros. Por lo tanto los implementadores pueden elegir diseñar sus implementaciones del AES con flexibilidad en mente hacia el futuro.

Para ambos, su cifrado y su cifrado inverso, el algoritmo AES hace uso de una función de ronda compuesta por cuatro diferentes transformaciones orientadas al nivel de bytes que serán descritas en el siguiente apartado.

4.4. Las Etapas

Ahora describiremos los pasos en más detalle. Como ya vimos, los 128 bits de entrada son agrupados en 16 bytes de 8 bits cada uno, denotados por

$$a_{0,0}, a_{1,0}, a_{2,0}, a_{3,0}, a_{0,1}, a_{1,1}, \ldots, a_{3,3}$$

Estos son ordenados en una matriz de tamaño 4×4

$$\begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

Esta matriz representa el arreglo de estados, la llamaremos *la matriz de estados*, cuyas entradas son bytes representados como elementos del campo de Rijndael $\mathbb{F} = \mathrm{GF}(2^8)$ anteriormente descrito.

Después de una ronda inicial de simple adición de la clave, el arreglo de estados es transformado implementando una función de ronda 10, 12 o 14 veces (dependiendo de la longitud de la clave), con la ronda final ligeramente diferente de las $Nr - 1$ rondas previas.

4.4.1. SubBytes

En esta etapa, cada uno de los bytes en la matriz de estados es reemplazado por otro byte a través de la siguiente tabla, llamada S-box.

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-	**63**	**7C**	**77**	**7B**	**F2**	**6B**	**6F**	**C5**	**30**	**01**	**67**	**2B**	**FE**	**D7**	**AB**	**76**
1-	**CA**	**82**	**C9**	**7D**	**FA**	**59**	**47**	**F0**	**AD**	**D4**	**A2**	**AF**	**9C**	**A4**	**72**	**C0**
2-	**B7**	**FD**	**93**	**26**	**36**	**3F**	**F7**	**CC**	**34**	**A5**	**E5**	**F1**	**71**	**D8**	**31**	**15**
3-	**04**	**C7**	**23**	**C3**	**18**	**96**	**05**	**9A**	**07**	**12**	**80**	**E2**	**EB**	**27**	**B2**	**75**
4-	**09**	**83**	**2C**	**1A**	**1B**	**6E**	**5A**	**A0**	**52**	**3B**	**D6**	**B3**	**29**	**E3**	**2F**	**84**
5-	**53**	**D1**	**00**	**ED**	**20**	**FC**	**B1**	**5B**	**6A**	**CB**	**BE**	**39**	**4A**	**4C**	**58**	**CF**
6-	**D0**	**EF**	**AA**	**FB**	**43**	**4D**	**33**	**85**	**45**	**F9**	**02**	**7F**	**50**	**3C**	**9F**	**A8**
7-	**51**	**A3**	**40**	**8F**	**92**	**9D**	**38**	**F5**	**BC**	**B6**	**DA**	**21**	**10**	**FF**	**F3**	**D2**
8-	**CD**	**0C**	**13**	**EC**	**5F**	**97**	**44**	**17**	**C4**	**A7**	**7E**	**3D**	**64**	**5D**	**19**	**73**
9-	**60**	**81**	**4F**	**DC**	**22**	**2A**	**90**	**88**	**46**	**EE**	**B8**	**14**	**DE**	**5E**	**0B**	**DB**
A-	**E0**	**32**	**3A**	**0A**	**49**	**06**	**24**	**5C**	**C2**	**D3**	**AC**	**62**	**91**	**95**	**E4**	**79**
B-	**E7**	**C8**	**37**	**6D**	**8D**	**D5**	**4E**	**A9**	**6C**	**56**	**F4**	**EA**	**65**	**7A**	**AE**	**08**
C-	**BA**	**78**	**25**	**2E**	**1C**	**A6**	**B4**	**C6**	**E8**	**DD**	**74**	**1F**	**4B**	**BD**	**8B**	**8A**
D-	**70**	**3E**	**B5**	**66**	**48**	**03**	**F6**	**0E**	**61**	**35**	**57**	**B9**	**86**	**C1**	**1D**	**9E**
E-	**E1**	**F8**	**98**	**11**	**69**	**D9**	**8E**	**94**	**9B**	**1E**	**87**	**E9**	**CE**	**55**	**28**	**DF**
F-	**8C**	**A1**	**89**	**0D**	**BF**	**E6**	**42**	**68**	**41**	**99**	**2D**	**0F**	**B0**	**54**	**BB**	**16**

*Figura 4. Tabla de sustitución S-box para el byte **XY** en formato hexadecimal.*

Dado un byte de 8 bits $\{\texttt{abcdefgh}\}$ se busca la entrada en la posición correspondiente a la intersección de la fila `abcd`$-$ (llevada a notación hexadecimal) y la columna $-$`efgh` (llevada a notación hexadecimal). Este valor expresado en notación binaria, se convierte en el byte de salida. Por ejemplo, si el byte de entrada es $\{\texttt{10001011}\}$, miramos la intersección de la fila `1000`$-$ (**8**$-$ en notación hexadecimal) y la columna $-$`1011` ($-$**B** en notación hexadecimal). El valor es **3D**, que es $\{\texttt{00111101}\}$ en notación binaria. Este es el byte salida de la S-box.

La salida de la etapa **SubBytes** es también una matriz 4×4 de bytes representada por

$$(b_{i,j}) = \begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix}.$$

4.4.2. ShiftRows

Las cuatro filas de la matriz son desplazadas cíclicamente hacia la izquierda tantas posiciones como indica el índice de cada fila, es decir 0, 1, 2, y 3 posiciones, para obtener

$$(c_{i,j}) = \begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,1} & b_{1,2} & b_{1,3} & b_{1,0} \\ b_{2,2} & b_{2,3} & b_{2,0} & b_{2,1} \\ b_{3,3} & b_{3,0} & b_{3,1} & b_{3,2} \end{pmatrix}.$$

Se ve con claridad que la primera fila permanece inalterada. La transformación para las demás filas está dada por:

$$c_{i,j} = b_{i,(j+shift(i,Nb)) \mod Nb}, \text{ para } 0 < i < 4 \text{ y } 0 \leq j < Nb,$$

donde $shift(i, Nb)$ depende de el número de fila i y esta definido por

$$shift(1, 4) = 1, shift(2, 4) = 2 \text{ y } shift(3, 4) = 3.$$

Esta expresión puede resumirse en la expresión equivalente:

$$b_{i,j} \mapsto b_{i,(j-i) \mod 4}$$

4.4.3. MixColumns

Considérese un byte como un elemento de GF (2^8). Entonces la salida de la etapa **ShiftRows** es una matriz $(c_{i,j})$ de tamaño 4×4 con valores en GF (2^8). Se multiplica ésta por la matriz $(m_{i,j})$

$$(m_{i,j}) = \begin{pmatrix} 00000010 & 00000011 & 00000001 & 00000001 \\ 00000001 & 00000010 & 00000011 & 00000001 \\ 00000001 & 00000001 & 00000010 & 00000011 \\ 00000011 & 00000001 & 00000001 & 00000010 \end{pmatrix} = \begin{pmatrix} \mathbf{02} & \mathbf{03} & \mathbf{01} & \mathbf{01} \\ \mathbf{01} & \mathbf{02} & \mathbf{03} & \mathbf{01} \\ \mathbf{01} & \mathbf{01} & \mathbf{02} & \mathbf{03} \\ \mathbf{03} & \mathbf{01} & \mathbf{01} & \mathbf{02} \end{pmatrix}$$

conformada también por elementos de GF (2^8), para producir la matriz de salida $(d_{i,j})$, como sigue

$$(m_{i,j}) \cdot (c_{i,j}) = \begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} \\ d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} \\ d_{3,0} & d_{3,1} & d_{3,2} & d_{3,3} \end{pmatrix} = (d_{i,j}).$$

La transformación **MixColumns** opera columna a columna sobre la matriz de estados, tratando cada columna como un polinomio de cuatro términos con coeficientes en GF (2^8) y multiplicándolas módulo $x^4 + 1$ por el polinomio fijo $a(x)$ dado por

$$a(x) = \{\mathbf{03}\}x^3 + \{\mathbf{01}\}x^2 + \{\mathbf{01}\}x + \{\mathbf{02}\}$$

que ya se vió en la subsección 4.2.1 y que allá mismo, quedó señalado que tal producto de polinomios puede ser escrito como una multiplicación de matrices. La expresión matricial dada arriba se sigue de aquélla descripción.

4.4.4. AddRoundKey

Las claves de ronda se derivan de la clave inicial principal siguiendo un proceso que se describirá luego. La clave de ronda consiste de Nb palabras (columnas) de 32 bits bits dispuestas en una matriz $(k_{i,j})$ de tamaño 4×4. Esta es *la matriz de la clave de ronda* y es adicionada mediante la operación XOR con la matriz de salida de la etapa **MixColumns**

$$\begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ d_{1,0} & d_{1,1} & d_{1,2} & d_{1,3} \\ d_{2,0} & d_{2,1} & d_{2,2} & d_{2,3} \\ d_{3,0} & d_{3,1} & d_{3,2} & d_{3,3} \end{pmatrix} \oplus \begin{pmatrix} k_{0,0} & k_{0,1} & k_{0,2} & k_{0,3} \\ k_{1,0} & k_{1,1} & k_{1,2} & k_{1,3} \\ k_{2,0} & k_{2,1} & k_{2,2} & k_{2,3} \\ k_{3,0} & k_{3,1} & k_{3,2} & k_{3,3} \end{pmatrix} = \begin{pmatrix} e_{0,0} & e_{0,1} & e_{0,2} & e_{0,3} \\ e_{1,0} & e_{1,1} & e_{1,2} & e_{1,3} \\ e_{2,0} & e_{2,1} & e_{2,2} & e_{2,3} \\ e_{3,0} & e_{3,1} & e_{3,2} & e_{3,3} \end{pmatrix} = (e_{i,j}).$$

Ésta es la salida final de la ronda. De manera formal, las Nb palabras de cada clave de ronda son adicionadas a las columnas del estado del modo siguiente

$$[e_{0,j}, e_{1,j}, e_{2,j}, e_{3,j}] = [d_{0,j}, d_{1,j}, d_{2,j}, d_{3,j}] \oplus [k_{i,r \cdot Nb+j}] \text{ para } 0 \leq i \leq 3 \text{ y } 0 \leq c < Nb,$$

donde r es un valor en el rango $1 \leq r \leq Nr$. Ésto se debe a que en el AES, la ronda inicial contempla sólo la adición vía XOR de la clave inicial.

Equivalentemente tenemos que en la ronda r, **AddRoundKey** puede ser expresada por $d_{i,j} \mapsto d_{i,j} \oplus k_{i,4r+i}$ con $0 \leq i, j \leq 3$

4.4.5. Programa de Expansión de la Clave

El algoritmo AES toma la clave principal K que consiste de 128, 192 o 256 bits divididos en bytes y dispuestos en un arreglo de tamaño $4 \times Nk$ y aplica una rutina de expansión que genera $Nb(Nr + 1)$ columnas. El algoritmo requiere un conjunto inicial de Nb columnas que conformarán la matriz de la clave inicial y cada una de las siguientes Nr rondas requiere Nb columnas de datos, que representarán las matrices de clave de cada ronda. Tales columnas se denotarán por $[w_i]$, con $0 \leq i < Nb(Nr + 1)$.

Etiquetamos las Nk primeras columnas de la matriz de la clave inicial por $W(0)$, $W(1)$, ..., $W(Nk)$. Las nuevas columnas son generadas recursivamente. Supongamos que las columnas hasta $W(i-1)$ ya han sido definidas. Si i no es múltiplo de 4, entonces

$$W(i) = W(i-4) \oplus W(i-1).$$

Si i es un múltiplo de $Nk = 4$, entonces

$$W(i) = W(i-4) \oplus T(W(i-1)).$$

donde $T(W(i-1))$ es la transformación de $W(i-1)$ obtenida siguiendo el siguiente procedimiento. Sean los elementos de la columna $W(i-1)$, $[k_{0,i-1}, k_{1,i-1}, k_{2,i-1}, k_{3,i-1}]$.

Movemos cíclicamente estos para obtener $[k_{1,i-1}, k_{2,i-1}, k_{3,i-1}, k_{0,i-1}]$, esta operación se denota como `RotWord`. Ahora reemplazamos cada uno de estos bytes con el correspondiente elemento en la S-box, obteniendo 4 bytes $[K_{0,i}, K_{1,i}, K_{2,i}, K_{3,i}]$, esta operación es conocida como `SubWord`. El resultado es sumado mediante la operación XOR, con la columna que se encuentra Nk posiciones antes, es decir $W(i - Nk)$, denotemos éste resultado por $[\kappa_{0,i}, \kappa_{1,i}, \kappa_{2,i}, \kappa_{3,i}]$. Finalmente se realiza una nueva suma XOR de esta última columna con la constante de ronda calculada por medio de la fórmula

$$\texttt{Rcon}(i) = \{\texttt{00000010}\}^{(i-4)/4} = \{\textbf{02}\}^{(i-4)/4}$$

en GF (2^8) (recuérdese que estamos en el caso en que i es múltiplo de 4). Entonces $T(W(i-1))$ es el vector columna

$$(\kappa_{0,i} \oplus \texttt{Rcon}(i), \kappa_{1,i}, \kappa_{2,i}, \kappa_{3,i}).$$

De esta manera, las columnas $W(Nk), \ldots, W(4Nr + 3)$ son producidas a partir de las columnas de la clave principal.

La clave de ronda para la r-ésima ronda está formada por la matriz cuyas columnas son

$$W(4r), W(4r + 1), W(4r + 2), W(4r + 3).$$

Sin embargo, es importante notar que la rutina de expansión de la clave para el AES-256 donde $Nk = 8$ es sutilmente diferente a las implementadas para AES-128 y AES-192. Si $Nk = 8$ e $i - 4$ es múltiplo de Nk, entonces `Subword` es aplicada a $W(i - 1)$ previamente a la operación XOR.

4.4.6. Construcción de la S-box

Aunque la S-box fue presentada como una tabla de resultados, esta tiene una simple descripción matemática. Comencemos con un byte $\{\texttt{x}_7\texttt{x}_6\texttt{x}_5\texttt{x}_4\texttt{x}_3\texttt{x}_2\texttt{x}_1\texttt{x}_0\}$, donde cada $\texttt{x}_i$ es un bit binario (0 o 1). Calculamos su inverso multiplicativo en GF (2^8) (esto se hace utilizando el algoritmo Euclideano extendido). Si el byte es $\{\texttt{00000000}\}$, no existe inverso, así que le haremos corresponder $\{\texttt{00000000}\}$ como inversa. El byte resultante representa un vector columna 8-dimensional $(y_7, y_6, y_5, y_4, y_3, y_2, y_1, y_0)$. Éste se multiplica por una matriz que representa una transformación lineal determinada, para después adicionar el vector columna $(1, 1, 0, 0, 0, 1, 1, 0)^T$ que representará la constante $\{\textbf{63}\} = \{\texttt{01100011}\}$ y así obtener un vector $(z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7)$ como sigue:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \end{pmatrix}.$$

El byte $\{z_7z_6z_5z_4z_3z_2z_1z_0\}$ es una posición en la S-box.

La transformación lineal descrita, es una transformación de $\mathrm{GF}(2^8)$ lineal sobre $\mathrm{GF}(2)$, y juntamente con la adición de la constante constituye una transformación lineal afín que tiene además la siguiente expresión

$$z_i = y_i \oplus y_{(i+4) \mod 8} \oplus y_{(i+5) \mod 8} \oplus y_{(i+6) \mod 8} \oplus y_{(i+7) \mod 8} \oplus c_i,$$

donde $0 \leq i < 8$ y c_i es el i-ésimo bit del byte constante.

Por ejemplo comencemos con el byte {`11001011`}. Su inverso en $\mathrm{GF}(2^8)$ es el byte {`00000100`}. Ahora calculamos

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Así, resulta el byte {`00011111`}.

Para verificar este resultado partamos del byte {`11001011`} y empleemos la S-box. Tomamos los cuatro primeros bits `1100`$-$ (**`C`**$-$ en notación hexadecimal) y los últimos cuatro bits $-$`1011` ($-$**`B`** en notación hexadecimal). Entonces buscamos el número en la posición correspondiente a la columna **`C`** y la fila **`B`** en la S-box. La entrada es **`1F`**, que en notación binaria es {`00011111`}.

Algunas de las consideraciones en el diseño de la S-box fueron las siguientes. La aplicación $x \mapsto x^{-1}$ fue usada para conseguir la propiedad de no-linealidad. Sin embargo, la simplicidad de tal aplicación podría posibilitar que el cifrado sea sensible a ciertos ataques y para evitar ésto, se combinó con la multiplicación por una matriz y la adición de un vector, como se describió previamente. La matriz es una matriz circular y fue elegida principalmente por su simplicidad (note que cada fila desde la segunda es un desplazamiento de la fila superior). El vector fue elegido de tal manera que: primero, el byte que se ingresa no sea igual al que devuelve la matriz; segundo, el byte que se ingresa no sea el complemento del que devuelve la matriz (El complemento de un byte se consigue cambiando cada 1 por 0 y cada 0 por 1).

4.5. Desencriptación

Cada una de las etapas **`SubBytes`**, **`ShiftRows`**, **`MixColumns`** y **`AddRoundKey`** son inversibles:

1. La inversa de **`SubBytes`** está dada por otra tabla de sustitución, llamada **`InvSubBytes`**.

	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-	**52**	**09**	**6A**	**D5**	**30**	**36**	**A5**	**38**	**BF**	**40**	**A3**	**9E**	**81**	**F3**	**D7**	**FB**
1-	**7C**	**E3**	**39**	**82**	**9B**	**2F**	**FF**	**87**	**34**	**8E**	**43**	**44**	**C4**	**DE**	**E9**	**CB**
2-	**54**	**7B**	**94**	**32**	**A6**	**C2**	**23**	**3D**	**EE**	**4C**	**95**	**OB**	**42**	**FA**	**C3**	**4E**
3-	**08**	**2E**	**A1**	**66**	**28**	**D9**	**24**	**B2**	**76**	**5B**	**A2**	**49**	**6D**	**8B**	**D1**	**25**
4-	**72**	**F8**	**F6**	**64**	**86**	**68**	**98**	**16**	**D4**	**A4**	**5C**	**CC**	**5D**	**65**	**B6**	**92**
5-	**6C**	**70**	**48**	**50**	**FD**	**ED**	**B9**	**DA**	**5E**	**15**	**46**	**57**	**A7**	**8D**	**9D**	**84**
6-	**90**	**D8**	**AB**	**00**	**8C**	**BC**	**D3**	**0A**	**F7**	**E4**	**58**	**05**	**B8**	**B3**	**45**	**06**
7-	**D0**	**2C**	**1E**	**8F**	**CA**	**3F**	**0F**	**02**	**C1**	**AF**	**BD**	**03**	**01**	**13**	**8A**	**6B**
8-	**3A**	**91**	**11**	**41**	**4F**	**67**	**DC**	**EA**	**97**	**F2**	**CF**	**CE**	**F0**	**B4**	**E6**	**73**
9-	**96**	**AC**	**74**	**22**	**E7**	**AD**	**35**	**85**	**E2**	**F9**	**37**	**E8**	**1C**	**75**	**DF**	**6E**
A-	**47**	**F1**	**1A**	**71**	**1D**	**29**	**C5**	**89**	**6F**	**B7**	**62**	**0E**	**AA**	**18**	**BE**	**1B**
B-	**FC**	**56**	**3E**	**4B**	**C6**	**D2**	**79**	**20**	**9A**	**DB**	**C0**	**FE**	**78**	**CD**	**5A**	**F4**
C-	**1F**	**DD**	**A8**	**33**	**88**	**07**	**C7**	**31**	**B1**	**12**	**10**	**59**	**27**	**80**	**EC**	**5F**
D-	**60**	**51**	**7F**	**A9**	**19**	**B5**	**4A**	**0D**	**2D**	**E5**	**7A**	**9F**	**93**	**C9**	**9C**	**EF**
E-	**A0**	**E0**	**3B**	**4D**	**AE**	**2A**	**F5**	**B0**	**C8**	**EB**	**BB**	**3C**	**83**	**53**	**99**	**61**
F-	**17**	**2B**	**04**	**7E**	**BA**	**77**	**D6**	**26**	**E1**	**69**	**14**	**63**	**55**	**21**	**0C**	**7D**

*Figura 5. Tabla de sustitución S-box inversa para el byte **XY** en formato hexadecimal.*

2. La inversa de **ShiftRows** se obtiene desplazando las filas de la misma forma pero hacia la derecha, produciendo así **InvShiftRows**.

3. La inversa de **MixColumns** existe porque la matriz usada en **MixColumns** es invertible. La transformación **InvMixColumns** se consigue multiplicando por la matriz

$$\begin{pmatrix} 00001110 & 00001011 & 00001101 & 00001001 \\ 00001001 & 00001110 & 00001011 & 00001101 \\ 00001101 & 00001001 & 00001110 & 00001011 \\ 00001011 & 00001101 & 00001001 & 00001110 \end{pmatrix} = \begin{pmatrix} \mathbf{0E} & \mathbf{0B} & \mathbf{0D} & \mathbf{09} \\ \mathbf{09} & \mathbf{0E} & \mathbf{0B} & \mathbf{0D} \\ \mathbf{0D} & \mathbf{09} & \mathbf{0E} & \mathbf{0B} \\ \mathbf{0B} & \mathbf{0D} & \mathbf{09} & \mathbf{0E} \end{pmatrix}.$$

4. **AddRoundKey** es su propia inversa.

La encriptación vía Rijndael consiste de los pasos siguientes (con las obvias abreviaciones)

ARK
SB, SR, MC, ARK
...
SB, SR, MC, ARK
SB, SR, ARK

Recordemos que MC se omite en la última ronda.

Para desencriptar, es necesario recorrer las inversas de estas etapas en el orden opuesto, originando el siguiente esquema preliminar de desencriptación:

ARK, ISR, ISB
ARK, IMC, ISR, ISB
...
ARK, IMC, ISR, ISB
ARK.

Sin embargo, es deseable que este esquema de desencriptación se parezca, tanto como se pueda, al esquema de encriptación.

Observe que aplicar SB y luego SR proporciona el mismo resultado que primero aplicar SR y luego SB. Ésto sucede así porque SB es una aplicación que actúa sobre un byte a la vez y SR permuta los bytes. Correspondientemente el orden de ISR e ISB también puede invertirse.

Resulta deseable que sea posible revertir el orden de ARK e IMC; pero ésto no es posible. En lugar de ello, procederemos aplicando MC y luego ARK a una matriz $(c_{i,j})$, ésto está dado por

$$(c_{i,j}) \longrightarrow (m_{i,j})(c_{i,j}) \longrightarrow (e_{i,j}) = (m_{i,j})(c_{i,j}) \oplus (k_{i,j}),$$

donde $(m_{i,j})$ de tamaño 4×4 corresponde a la matriz presente en la operación **`MixColumns`** y $(k_{i,j})$ es la matriz de la clave de ronda. La inversa se obtiene resolviendo la ecuación $(e_{i,j}) = (m_{i,j})(c_{i,j}) \oplus (k_{i,j})$ para $(c_{i,j})$ en términos de $(e_{i,j})$, ésto es $(c_{i,j}) = (m_{i,j})^{-1}(e_{i,j}) \oplus (m_{i,j})^{-1}(k_{i,j})$. Por lo tanto el proceso es

$$(e_{i,j}) \longrightarrow (m_{i,j})^{-1}(e_{i,j}) \longrightarrow (m_{i,j})^{-1}(e_{i,j}) \oplus (k'_{i,j}),$$

donde $(k'_{i,j}) = (m_{i,j})^{-1}(k_{i,j})$. La primera flecha se interpreta simplemente como **`InvMixColumns`** aplicado a $(e_{i,j})$. Si hacemos que **`InvAddRoundKey`** represente adición XOR con $(k'_{i,j})$, entonces tendremos que la inversa de "MC luego ARK"será "IMC luego IARK". Por lo tanto, podemos reemplazar los pasos "ARK luego IMC"por los pasos "IMC luego IARK"en el precedente esquema descriptivo. Vemos así que el esquema de desencriptación queda como

ARK, IBS, ISR
IMC, IARK, IBS, ISR
...
IMC, IARK, IBS, ISR
ARK.

Reagrupando las líneas obtendremos la versión final:

Desencriptación vía el Algoritmo AES	
Ronda	Vinculación de Etapas
0	ARK, usando la clave 10.
1	IBS, ISR, IMC, IARK, usando la clave 9.
2	IBS, ISR, IMC, IARK, usando la clave 8.
⋮	⋮
9	IBS, ISR, IMC, IARK, usando la clave 1.
10	IBS, ISR, ARK, usando la clave inicial.

Por lo tanto, la desencriptación tiene esencialmente la misma estructura que la encriptación con **`SubBytes`**, **`ShiftRows`** y **`MixColumns`** siendo reemplazadas por sus inversas y **`AddRoundKey`** por su inversa **`InvAddRoundKey`**, excepto en las rondas inicial y final. Desde luego, las claves de ronda son usadas en el orden opuesto, así el primer ARK hace uso de la décima clave de ronda y el último ARK emplea la clave de ronda inicial.

Lo precedente muestra por qué **`MixColumns`** es omitido en la última ronda. Supongamos que MC permanece en la ronda, entonces la encriptación comenzaría con ARK, SB, SR, MC, ARK,..., y finalizaría en ARK, SB, SR, MC, ARK. En consecuencia la desencriptación comenzaría, después de la reordenación respectiva, con IMC, IARK, ISB, ISR,... Esto significa que la desencriptación tiene un innecesario paso IMC al inicio que tendría el efecto de retardar el algoritmo.

Otra forma de visualizar el proceso de encriptación consiste en iniciar con ARK y de aquí se sigue una secuencia alternada de medias rondas

(SB, SR), (MC, ARK), (SB, SR),..., (MC, ARK), (SB, SR),

seguida por una ARK final. La desencriptación es dada por una ARK seguida por una secuencia alternada de medias rondas

(ISB, ISR), (IMC, IARK), (ISB, ISR),..., (IMC, IARK), (ISB, ISR),

seguida por un ARK final. Desde este punto de vista, es notable que una MC final no cabe efectivamente en ninguna de las medias rondas, y resulta natural mantenerla fuera.

En procesadores de 8 bits, el proceso de desencriptación no es lo bastante rápida como el de encriptación. Esto es debido a que las entradas en la matriz de **`InvMixColumns`** son más complejas que aquéllas de **`MixColumns`**, y esto basta para hacer que la desencriptación tome alrededor de un $30\,\%$ más tiempo de procesamiento que la encriptación para estos procesadores. Sin embargo, en muchas aplicaciones, la desencriptación no es necesaria, por ejemplo, cuando el algoritmo opera en modo CFB (Cipher Feedback), donde un byte es encriptado sin tener que esperar hasta que un bloque entero esté disponible. Por lo tanto, no se tiene acá una significativa desventaja.

El hecho de que la encriptación y la desencriptación no sean procesos idénticos, conduce a la expectativa de que no hay claves débiles, en contraste a DES y otros algoritmos.

4.6. Consideraciones de Diseño

El diseño del algoritmo Rijndael está en concordancia con la estrategia del rastro amplio. Este tópico no será objeto de estudio del presente trabajo, por lo cual se sugiere su consulta en [1] y en [10]. En ambos se hace un exhaustivo tratamiento del tema. En [10] se hace una exposición un poco más formal, no obstante al ser [1] la obra de Joan Daemen y Vincent Rijmen, los creadores y diseñadores del algoritmo Rijndael es preciso leer éste material como punto de partida.

El algoritmo Rijndael no es un Sistema Feistel. En un sistema Feistel, la mitad de los bits son movidos pero no cambiados durante cada ronda. En Rijndael, todos los bits son tratados uniformemente. Esto tiene el efecto de difuminar rápidamente la información contenida en los bits de entrada. Y se puede mostrar que dos rondas son suficientes para obtener una difusión completa, es decir, que cada uno de los 128 bits de salida depende de cada uno de los 128 bits de entrada.

Cada componente del AES fue cuidadosamente elegida y tiene un específico rol. Cada ronda está dividida en tres partes. La primera es **`SubBytes`** y es llamada el *estrato de sustitución*. La segunda parte la conforman **`ShiftRows`** seguida de **`MixColumns`**. Esta parte recibe el nombre de *estrato de difusión*. La parte final introduce el material de la clave a través de **`AddRoundKey`**. Discutamos ahora sobre ambos estratos.

4.6.1. Estrato de Sustitución

Esta capa está basada en la S-box del AES que a su vez, está definida por una composición de tres operaciones: La inversión extendida al elemento cero; La aplicación GF(2)-lineal de GF(2^8) especificada por la matriz circulante presentada en la subsección 4.4.6 y la adición de la constante **`63`**

La S-box fue construida en una simple y explícita forma álgebraica, esto para eliminar cualquier sospecha de puertas trampa. Este deseo de eliminar los misterios sobre las S-box fue lo que terminó inhabilitando al DES. La S-box de Rijndael es altamente no lineal, porque se basa en la aplicación $x \mapsto x^{-1}$ en GF(2^8). Es excelente resistiendo criptoanálisis lineal y diferencial, así como los más recientes métodos criptoanalíticos llamados ataques de interpolación. La razón para el empleo de la aplicación GF(2)-lineal y la constante de la S-box, es la de incrementar la complejidad algebraica de la S-box y remover puntos fijos respectivamente.

4.6.2. Estrato de Difusión

Para el AES, la estrategia del rastro amplio está basada en la matriz de 4×4 sobre $\mathbb{F}$ utilizada en **`MixColumns`**. Ésta, es la matriz de comprobación de paridad para un código MDS (Maximal Distance Separable) y es conocida como una matriz MDS (definida por la propiedad de que cada una de sus submatrices cuadradas es invertible). Sabemos que una matriz de 4×4 sobre $\mathbb{F}$ opera sobre 4 bytes de entrada y genera 4 bytes de salida. Para la matriz MDS de **`MixColumns`**, se tiene que, o bien que todos los bytes de entrada y salida son cero o, al menos 5 de estos 8 bytes son diferentes de cero. Esta propiedad MDS es útil para asegurar que el número de S-box activas involucradas en un ataque lineal o diferencial, se incrementará rápidamente y la seguridad del AES contra estos ataques en particular estará garantizada.

El paso **`ShiftRows`** fue adicionado para resistir dos recientes ataques desarrollados, denominados truncado diferencial y ataque Square (Square fue un predecesor de Rijndael). Mientras tanto, el paso **`MixColumn`** provoca la difusión sobre los bytes. Un cambio en un byte entrada en este paso, siempre resulta en cambios en los cuatro bytes de salida. Si dos bytes de entrada son cambiados, entonces al menos tres bytes de salida cambiaran.

4.6.3. Estrato de Incorporación de la Clave

La programación de claves involucra un efecto de mezcla no lineal de los bits de la clave dado que hace uso de la S-box. El proceso está diseñado para resistir ataques donde los criptoanalistas conocen parte de la clave e intentan deducir los bits restantes. También, tiene como objetivo asegurar que dos claves distintas no tengan un gran número de claves de ronda en común. Las constantes de ronda son usadas para eliminar simetrías en el proceso de encriptación haciendo cada ronda diferente.

Capítulo 5

Propiedades Algebraicas del AES

Los primeros comentarios públicos sobre la estructura algebraica del algoritmo Rijndael fueron hechos hacia la conclusión del proceso de selección del AES. Este capítulo proveerá un sumario de muchos de los trabajos al respecto y que siguieron a la publicación del algoritmo Rijndael ya en calidad de estándar.

5.1. Estructura de Ronda

El diseño del AES es un ejemplo de una SP-network (substitution-permutation network, una concatenación de transformaciones que pueden verse como operaciones de sustitución o permutación), en la que cada ronda usualmente consiste de tres fases. La primera fase es una transformación no lineal localizada o sustitución del estado, ésto quiere decir que transformaciones no lineales son aplicadas a los diferentes sub-bloques del estado. La segunda fase es una extensiva difusión lineal del estado integro. Y la fase final combina el estado con el material de la clave. De acuerdo con el fundamento lógico de diseño del AES, la primera fase es ejecutada por la operación **`SubBytes`**; la segunda fase por la combinación de las operaciones **`ShiftRows`** y **`MixColumns`**; y la fase final por la operación **`AddRoundKey`**.

El rol de **`ShiftRows`** y **`MixColumns`** es el de proveer difusión en el AES. Ambas son transformaciones lineales del estado del cifrado sobre el campo de Rijndael $\mathbb{F}$. La operación **`ShiftRows`** proporciona lo que se conoce como *elevada dispersión*, mientras que la operación **`MixColumns`** suministra una *elevada difusión local*. Las dos operaciones se combinan para provocar una alta y eficaz difusión, conforme a los requerimientos de la estrategia del rastro amplio. No obstante, en esta sección se propone un método alternativo de análisis para la difusión en el AES. Este análisis emplea simples herramientas algebraicas con el propósito de explorar la estructura subyacente de las operaciones que componen el AES, así como de sus combinaciones. Comenzaremos por considerar las operaciones en una simple ronda del AES.

5.1.1. La Operación SubBytes

La S-box del AES está compuesta por tres transformaciones llamadas: la inversión aumentada $w \mapsto w^{-1}$, una aplicación $\mathrm{GF}(2)$-lineal y la adición de una constante. La operación de inversión tiene propiedades que resisten el criptoanálisis estándar, mientras que las otras transformaciones en la S-box disfrazan la simplicidad algebraica de aquélla y así proporcionan una "complicada expresión algebraica siempre que esté combinada con la aplicación inversa". De aquí se tiene ya un argumento en el sentido de que el AES sería resistente a los ataques de interpolación y otros similares. Además dicho sea de paso, la constante **63** en la S-box fue "elegida de tal manera que la S-box no tenga puntos fijos ni puntos fijos opuestos".

Las operaciones finales de la S-box, la aplicación $\mathrm{GF}(2)$-lineal, y la adición de una constante **63**, conforman una transformación lineal afín sobre $\mathrm{GF}(2)$.

5.1.2. Las Operaciones ShiftRows y MixColumns

La operación **ShiftRows** consiste en la rotación de las filas de la matriz de estado. La rotación de una fila en una posición es representada por la matriz de permutación $\widehat{R}$ de 4×4 sobre $\mathbb{F}$, donde

$$\widehat{R} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Si cambiamos las bases del espacio de estados de tal forma que el arreglo de estados esté representado por el vector columna

$$(\mathcal{S}_0, \mathcal{S}_4, \mathcal{S}_8, \mathcal{S}_{12}, \mathcal{S}_1, \mathcal{S}_5, \mathcal{S}_9, \mathcal{S}_{13}, \mathcal{S}_2, \mathcal{S}_6, \mathcal{S}_{10}, \mathcal{S}_{14}, \mathcal{S}_3, \mathcal{S}_7, \mathcal{S}_{11}, \mathcal{S}_{15})^T$$

entonces la acción que lleva a cabo la operación **ShiftRows** quedará representada por la matriz diagonal por bloques de 16×16

$$\begin{pmatrix} I & 0 & 0 & 0 \\ 0 & \widehat{R} & 0 & 0 \\ 0 & 0 & \widehat{R}^2 & 0 \\ 0 & 0 & 0 & \widehat{R}^3 \end{pmatrix}.$$

Reordenando las filas y las columnas de ésta matriz, podemos obtener una matriz $\overline{R}$ de 16×16 sobre $\mathbb{F}$ que representa la operación **ShiftRows** con respecto al ordenamiento estándar por columnas de la matriz de estado.

La operación **MixColumns** es típicamente descrita en términos de la matriz de 4×4 sobre $\mathbb{F}$ dada en el capítulo 4. Sin embargo, existe una base de $\mathbb{F}^4$ en la cual la transformación dada por dicha matriz puede ser representada usando la matriz $\widehat{R}$. Luego, existe una base de $\mathbb{F}^{16}$ tal que la operación **MixColumns** está dada por la matriz diagonal por bloques

$$\begin{pmatrix} \widehat{R} & 0 & 0 & 0 \\ 0 & \widehat{R} & 0 & 0 \\ 0 & 0 & \widehat{R} & 0 \\ 0 & 0 & 0 & \widehat{R} \end{pmatrix}.$$

La matriz $\overline{C}$ que representa la operación **MixColumns** con respecto al ordenamiento estándar del arreglo de estados tiene las mismas propiedades algebraicas que esta matriz.

La acción combinada de **ShiftRows** seguida por **MixColumns** queda representada por la matriz $\overline{C}\,\overline{R}$ de 16×16 sobre $\mathbb{F}$ cuya interpretación es la difusión lineal señalada por la estrategia del rastro amplio. Como $\overline{C}$ y $\overline{R}$ se basan esencialmente en R, podemos determinar las propiedades algebraicas de estas operaciones constituyentes y de sus combinaciones.

Consideremos el vector de estado, como un vector de longitud 16 sobre $\mathbb{F}$ o como un vector sobre $\mathrm{GF}(2)$ de longitud 128. En el último caso, las operaciones **ShiftRows** y **MixColumns** serán dadas por las matrices de 128×128 R y C respectivamente. La multiplicación por un elemento de $\mathbb{F}$ es una transformación lineal de $\mathbb{F}$ considerado como el espacio vectorial $\mathrm{GF}(2)^8$, entonces está descrita por una matriz 8×8 sobre $\mathrm{GF}(2)$. Luego, las matrices R y C estan dadas por matrices por bloques en las que las entradas 1, θ y $\theta + 1$ de $\overline{C}$ y $\overline{R}$ son reemplazadas por las matrices 8×8, I, T_θ y $T_{\theta+1}$ (ver el ejemplo dado en la sección 3.3). La difusión lineal requerida por la estrategia del rastro amplio estará por lo tanto descrita por la matriz CR de 128×128 sobre $\mathrm{GF}(2)$. Las propiedades algebraicas de R y C y por ende de CR, se desprenden de aquéllas propias de $\overline{R}$ y $\overline{C}$.

5.1.3. Difusión Lineal Aumentada

Ya vimos que las dos partes finales de la operación **SubBytes**, a saber la aplicación $\mathrm{GF}(2)$-lineal, y la adición de una constante **63**, forman una operación afín sobre $\mathrm{GF}(2)$. Además las operaciones de difusión del AES sobre los bytes del espacio de estados, **ShiftRows** y **MixColumns**, son también operaciones lineales sobre $\mathrm{GF}(2)$. Es por lo tanto razonable considerar una difusión lineal aumentada para el AES, que surge de la combinación de la transformación afín al interior de la operación **SubBytes** con las operaciones **ShiftRows** y **MixColumns**. De la combinación de estas operaciones en una, derivamos una muy natural descomposición matemática de la función de ronda del AES. La primera parte consiste de la inversión simultánea de todas las componentes del estado. La segunda parte es afín sobre $\mathrm{GF}(2)$ y consiste de la composición de todas las otras operaciones en la ronda.

La descomposición de la función de ronda de un cifrado por bloques en una parte lineal y una parte afín es un tanto arbitraria, ya que claramente existen muchas formas en las que tal descomposición puede realizarse. Aunque claro, dado que esta parte no lineal es particularmente simple, la división de la

función de ronda del AES en una parte no lineal de inversión y una parte afín consistente de las operaciones restantes, es sorprendentemente clara.

A continuación se dará una expresión de la parte afín de la función de ronda del AES, la cual no es más que la función de ronda, desprovista de la operación de inversión. Consideremos los vectores de longitud 128 sobre GF (2), y supongamos que x es el elemento de salida de la operación de inversión en la ronda, que $\mathbf{k}_i$ es la clave de ronda, y que **63** es el vector de las constantes generadas en cada aparición de la S-box. La parte afín de la aplicación de ronda está dada entonces por

$$\mathbf{x} \mapsto CR(L\mathbf{x} + \mathbf{63}) + \mathbf{k}_i$$

donde C, R y L son las matrices tratadas anteriormente. Considerando que $C(\mathbf{63}) = R(\mathbf{63}) = \mathbf{63}$, entonces la aplicación afín se puede escribir como

$$\mathbf{x} \mapsto CRL\mathbf{x} + \mathbf{k}_i + \mathbf{63}.$$

Así, la transformación lineal de esta aplicación afín está dada por la matriz $M = CRL$ de 128×128 sobre GF (2). La matriz M es particularmente simple y es ligeramente más compleja que la matriz de difusión lineal CR identificada por la estrategia del rastro amplio. Por lo tanto consideraremos la *difusión lineal aumentada* dada por la matriz M en nuestro subsecuente análisis del AES. Ésto nos permitirá expresar la función de ronda del AES de forma sucinta.

Ahora, se establecerá dicha expresión para la función de ronda del AES. Supongamos que w y x son vectores de longitud 128 sobre GF (2) y que w es la entrada y x es la salida de la operación de inversión. Entonces tenemos

$$\mathbf{x} = (x_0, \ldots, x_{15})^T = (w_0^{(-1)}, \ldots, w_{15}^{(-1)})^T = \mathbf{w}^{(-1)},$$

donde $\mathbf{w}^{(-1)}$ denota la inversión en cada componente. Asimismo, podemos definir un esquema revisado de generación de claves para el AES con claves de ronda dadas por $\mathbf{k}_i^* = \mathbf{k}_i + \mathbf{63}$ (para $i > 0$) con $\mathbf{k}_0^* = \mathbf{k}_0$. Una ronda del AES está dada entonces por

$$\mathbf{w} \mapsto M\mathbf{w}^{(-1)} + \mathbf{k}_i^*.$$

Podemos por lo tanto considerar una definición equivalente de la función de ronda del AES en la que una S-box consiste solamente de la operación inversión.

En consecuencia, mientras que un criterio de diseño para la S-box es que este no tenga "puntos fijos", el equivalente de la S-box en la (algebraicamente) más simple descripción del AES tiene dos puntos fijos (**00** y **01**). La difusión en las ronda del AES está dada ahora por la difusión lineal aumentada, y una ronda del AES consiste solamente de las siguientes dos simples operaciones algebraicas.

- Una inversión en cada componente sobre el campo $\mathbb{F}$.
- Una transformación afín de un espacio vectorial sobre GF (2).

5.1.4. Propiedades de la Difusión Lineal Aumentada

Ahora discutiremos algunas de las propiedades básicas de la matriz de difusión lineal aumentada $M = CRL$ y sus matrices componentes. Detallamos estas propiedades en la siguiente tabla

	Matriz usada en la difusión aumentada				
	C	R	CR	L	M
Polinomio minimal	x^4+1	x^4+1	x^8+1	$(x+1)^3$	$(x+1)^{15}$
Orden	4	4	8	4	16
Dimensión: Subespacio fijo	32	64	16	48	16
Dimensión: Subespacio de orden 2	64	96	32	96	30
Dimensión: Subespacio de orden 4	128	128	64	128	58
Dimensión: Subespacio de orden 8	128	128	128	128	96

Figura 1. El subespacio de orden i para la matriz T es $\{\mathbf{v}|\ T^i\mathbf{v} = \mathbf{v}\}$.

En particular el polinomio minimal $\min_M(x)$ de M está dado por

$$\min_M(x) = (x+1)^{15},$$

y $\min_M(x)$ divide a $(x+1)^{16} = x^{16}+1$. Luego M tiene orden 16, lo cual significa que cualquier entrada de 128 bits a la difusión lineal aumentada del AES, será aplicada a si misma después de a lo sumo 16 aplicaciones repetidas de la difusión lineal aumentada. Que el orden sea tan pequeño sorprende porque ésto sugiere que la difusión lineal aumentada posee una estructura considerable, aún cuando esta incluye sólo dos de las tres partes de la altamente no lineal S-box. Además, la transformación afín A_k de la difusión aumentada dada por $\mathbf{x} \mapsto M\mathbf{x} + \mathbf{k}$ tiene orden 16 ya que

$$\begin{aligned} A_{\mathbf{k}}^{16}\mathbf{x} &= M^{16}\mathbf{x} + (M^{15} + M^{14} + \cdots + M + I)\mathbf{k} \\ &= I\mathbf{x} + (M+I)^{15}\mathbf{k} = \mathbf{x}. \end{aligned}$$

Propiedades adicionales de la matriz de difusión lineal aumentada pueden ser encontradas aplicando una transformación de cambio de base a la matriz M, sea P esta matriz. La simple estructura de la difusión lineal aumentada del AES revelada por $P^{-1}MP$ muestra 15 subespacios $V_1, \ldots, V_{15} \subset \mathrm{GF}(2)^{128}$ tales que

$$\mathrm{GF}(2)^{128} = V_1 \oplus V_2 \oplus \cdots \oplus V_{15}.$$

Estos subespacios $V_1, \ldots, V_{15}$ tienen dimensiones 16, 14, 14, 14, 10, 10, 10, 8, 8, 6, 4, 4, 4, 4 y 2 respectivamente. Ellos tienen la propiedad de que, si $v_i \in V_i$, entonces $Mv_i = v_i + v_{i-1}$ para algún $v_{i-1} \in V_{i-1}$ ($i > 1$) con $Mv_1 = v_1$. Además los subespacios $V'_j = V_1 \oplus \cdots \oplus V_j$ ($j = 1, \ldots, 15$) son M-invariantes, ésto es $MV'_j = V'_j$.

Tales propiedades de la difusión lineal aumentada del AES sugieren varias ideas para el análisis del algoritmo, algunas de las cuales son mencionadas

adelante. Las técnicas involucradas requieren entendimiento sobre la estructura y la construcción de estos subespacios invariantes.

Hemos visto ya que el subespacio 16-dimensional $V_1 = V_1'$ es fijado por la matriz M. Luego, hay 2^{16} vectores que quedan fijos por la difusión aumentada. Además suponiendo que $\mathbf{x}$ y $\mathbf{x}'$ son dos vectores tales que la diferencia $\mathbf{x} + \mathbf{x}' \in V_1$, entonces

$$M\mathbf{x} + M\mathbf{x}' = M(\mathbf{x} + \mathbf{x}') = \mathbf{x} + \mathbf{x}',$$

y así, la difusión aumentada del AES fija también 2^{16} diferencias. En particular, existen vectores que quedan fijos por M y son no nulos para sólo 12 de los 16 bytes del estado. De esta manera el uso de una tal diferencia en un análisis del AES involucraría solo 12 S-box activas en cada ronda. Uno de tales vectores sobre GF (2) dado en notación hexadecimal es

$$(\mathtt{55336600\ 33550066\ 55336600\ 33550066})^T.$$

Tal análisis de la matriz M de difusión aumentada se puede extender a los tests de paridad. En este caso, un *test de paridad* es un vector fila $\mathbf{e}^T$ de longitud 128 sobre GF (2), y el *valor comprobación de paridad* de un vector $\mathbf{x}$ es $\mathbf{e}^T\mathbf{x} \in$ GF (2). Además, existen 2^{16} vectores fila $\mathbf{e}^T$ que son fijados por la matriz M de difusión aumentada, esto es $\mathbf{e}^T M = \mathbf{e}^T$. Para tal test de paridad $\mathbf{e}^T$, cualquier valor de comprobación de paridad es fijado por la difusión aumentada como

$$\mathbf{e}^T M\mathbf{x} = \mathbf{e}^T\mathbf{x}.$$

De igual manera, cualquier valor de comprobación de paridad de una diferencia es también fijado por la difusión aumentada como

$$\mathbf{e}^T(M\mathbf{x} + M\mathbf{x}') = \mathbf{e}^T M(\mathbf{x} + \mathbf{x}') = \mathbf{e}^T(\mathbf{x} + \mathbf{x}').$$

Sin embargo, existen vectores fila de test de paridad fijos que solo tienen 12 bytes no nulos, y que por lo tanto involucran solo 12 S-box activas. Uno de tales vectores fila de test de paridad sobre GF (2) dado en notación hexadecimal es

$$(\mathtt{00999900\ CC5555CC\ 00999900\ CC5555CC}).$$

Existen muchas otras formas en que tales tests de paridad pueden ser utilizados en el análisis del AES. Por ejemplo, vimos anteriormente que el subespacio V_{14}' 126-dimensional es M-invariante. Además, cualquier clase de V_{14}' es aplicada a si misma por M. De esta manera tenemos identificada una partición ya sea del espacio de estados o del conjunto de diferencias, en cuatro subconjuntos en los cuales esta partición es preservada por la difusión lineal aumentada. Estas cuatro clases están definidas por los dos vectores fila de test de paridad sobre GF (2) dados en notación hexadecimal por

$$(\mathtt{AAAAAAAA\ AAAAAAAA\ AAAAAAAA\ AAAAAAAA}),$$
$$(\mathtt{5AF05AF0\ 5AF05AF0\ 5AF05AF0\ 5AF05AF0}).$$

Tomando $\mathbb{F}$ en lugar de GF (2) se ha evidenciado algunos efectos de la alta probabilidad diferencial bajo secuencias relacionadas de claves de ronda. Estas observaciones demuestran que tales efectos ser observados en algoritmos de encriptación tipo AES que cumplen con las demandas de la estrategia del rastro amplio, por lo cual un mayor análisis de estos temas podría arrojar luces en el estudio del AES.

5.2. Representaciones Algebraicas

Puede haber una variedad de formas equivalentes de describir un criptosistema. Si bien la estandarización requiere la misma convención para su uso en la representación de datos, representaciones alternativas de las operaciones del cifrado pueden ser de mucho interés. Algunas representaciones pueden ser muy útiles en la implementación, quizá para mejorar el desempeño o para proveer protección adicional contra ciertos tipos de ataques. Otras representaciones pueden ser útiles para el criptoanálisis en la espera de que las mismas, revelen propiedades adicionales del criptosistema.

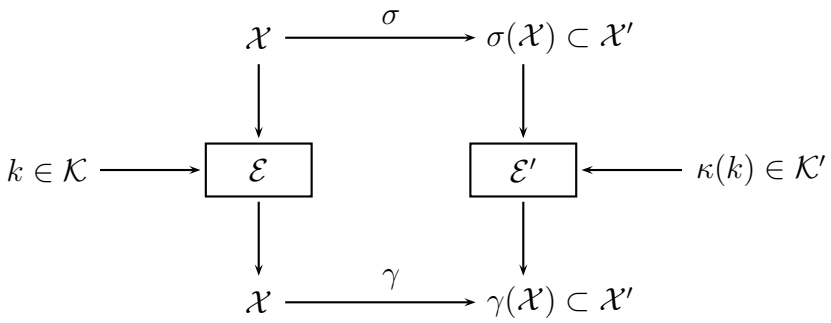

Figura 2. $\mathcal{E}'$: Una representación alternativa del cifrado $\mathcal{E}$.

Representaciones alternativas para cifrados por bloques son construidas definiendo aplicaciones. Supongamos que tenemos un cifrado por bloques original $\mathcal{E}$ con un espacio de estados $\mathcal{X}$ y un espacio de claves $\mathcal{K}$ y un nuevo cifrado por bloques $\mathcal{E}'$ con un espacio de estados $\mathcal{X}'$ y un espacio de claves $\mathcal{K}'$. Ahora podemos definir una aplicación de texto-franco σ, una aplicación de texto-cifrado γ, y una aplicación de clave κ entre los respectivos espacios de los cifrados dados, de tal forma que

$$\sigma, \gamma : \mathcal{X} \to \mathcal{X}' \text{ y } \kappa : \mathcal{K} \to \mathcal{K}'.$$

Diremos que el cifrado por bloques $\mathcal{E}'$ es una *representación alternativa* del cifrado por bloques $\mathcal{E}$ si, para todo $x \in \mathcal{X}$ y $k \in \mathcal{K}$,

$$\mathcal{E}'_{\kappa(k)}(\sigma(x)) = \gamma(\mathcal{E}_k(x)).$$

Una representación alternativa es mejor ilustrada por el diagrama conmutativo presentado en la Figura 2. Si las aplicaciones son inyectivas, entonces podemos replicar la encriptación hecha a través de $\mathcal{E}$ usando el cifrado $\mathcal{E}'$. Asignamos el texto-franco original al nuevo texto-franco por medio de σ y asignamos

la clave original a la nueva clave por medio de κ. Entonces encriptamos el nuevo texto-franco con $\mathcal{E}'$ bajo la nueva clave para obtener un nuevo texto-cifrado. Podemos recuperar el texto-cifrado original a partir del nuevo texto-cifrado. El texto-cifrado recuperado es el que se habría obtenido si se hubiera encriptado directamente con el cifrado por bloques $\mathcal{E}$ original. En este caso, diremos que el cifrado $\mathcal{E}$ está *encajado* en el cifrado $\mathcal{E}'$.

Las técnicas criptoanalíticas para cifrados por bloques pueden a veces describirse con el empleo de diagramas conmutativos y sus generalizaciones. Es el caso por ejemplo de cualquier cifrado por bloques con *factores o estructuras lineales*, para los cuales una de tales técnicas puede plantearse y cuyas propiedades tienen el potencial de reducir el costo de búsqueda de la clave, utilizando encriptaciones algebraicamente relacionadas; pero con la consiguiente reducción en la complejidad de los cálculos.

Representaciones alternativas donde el cifrado original y el nuevo cifrado por bloques son idénticos ($\mathcal{X}' = \mathcal{X}$, $\mathcal{K}' = \mathcal{K}$ y $\mathcal{E}' = \mathcal{E}$) son denominadas *auto-duales*. La propiedad de un cifrado de ser auto-dual bajo cualquier aplicación afín no trivial es esencialmente equivalente a la propiedad de poseer factores o estructuras lineales.

EJEMPLO. La propiedad de complementación del DES da un cifrado auto-dual no trivial o, equivalentemente, una estructura lineal. Sean 1_x y 1_k que denotan los vectores $(1, \ldots, 1)^T$ de longitud 64 y 56 respectivamente. Entonces se toman σ y γ como la aplicación $x \mapsto x + 1_x$, y κ como la aplicación $k \mapsto k + 1_k$. Éstas son conocidas como aplicaciones de complementación y determinan un cifrado auto-dual para el DES. Alternativamente, podemos obtener una estructura lineal colocando

$$\mathcal{X}' = \frac{\mathcal{X}}{\langle 1_x \rangle} \text{ y } \mathcal{K}' = \frac{\mathcal{K}}{\langle 1_k \rangle}$$

para ser espacios cociente con sus respectivas aplicaciones naturales $\sigma = \gamma$ y κ. El diagrama conmutativo equivalente al de la Figura 2 puede ser completado colocando

$$\mathcal{E}'_{\kappa(k)}(\sigma(x)) = \mathcal{E}_{k+1_k}(x + 1_x) = \mathcal{E}_k(x) + 1_x.$$

EJEMPLO. La función de ronda del AES es auto-dual. Definamos φ como la permutación de los bytes del arreglo de estados o de la clave definido por

$$\varphi = (\mathcal{S}_{00}\mathcal{S}_{03}\mathcal{S}_{02}\mathcal{S}_{01})(\mathcal{S}_{10}\mathcal{S}_{13}\mathcal{S}_{12}\mathcal{S}_{11})(\mathcal{S}_{20}\mathcal{S}_{23}\mathcal{S}_{22}\mathcal{S}_{21})(\mathcal{S}_{30}\mathcal{S}_{33}\mathcal{S}_{32}\mathcal{S}_{31}).$$

Definamos las tres aplicaciones ρ, γ, y κ a ser, las aplicaciones de los espacios de estado y de clave inducidas por una de las permutaciones φ, φ^2, o φ^3 del arreglo de bytes. Si $\mathcal{E}$ denota la función de ronda del AES, entonces

$$\mathcal{E}_{\kappa(k)}(\sigma(x)) = \gamma(\mathcal{E}_k(x)),$$

así la función de ronda del AES es auto-dual. Sin embargo, esta propiedad no se puede extender a la totalidad del AES, debido a la acción de la expansión de la clave.

Un entorno tal de trabajo para representaciones alternativas, puede ser extendido estocásticamente. Las técnicas estadísticas estándar del criptoanálisis para cifrados por bloques, tal como el criptoanálisis diferencial y lineal, pueden ser descritas usando esta simple generalización. Simplemente vemos el diagrama conmutativo como sostenible estadísticamente y requerimos que la función $\mathcal{E}'$ complete el diagrama conmutativo con una probabilidad adecuada.

5.2.1. Representaciones del AES

Varias representaciones alternativas han sido propuestas para el AES. Ellas explotan la estructura del cifrado y en su mayoría son construidas definiendo homomorfismos del espacio de estados y el de claves del AES.

Como se vió en el capítulo 4, el espacio de estados del AES está formado por 16 bytes, donde cada byte es identificado con un elemento del campo de Rijndael $\mathbb{F}$. El conjunto $\mathbb{F}^{16}$ tiene una estructura de espacio vectorial y una estructura de anillo con la multiplicación componente a componente. Por lo tanto podemos considerar el espacio de estados del AES con esta estructura natural como un $\mathbb{F}$-álgebra que llamaremos *álgebra del espacio de estados del AES* o brevemente *álgebra de estados*.

Las transformaciones algebraicas del álgebra de estados, que son transformaciones que preservan gran parte de la estructura del álgebra, están basadas en o bien una transformación lineal o en su defecto en una transformación teórica de anillos, sobre el espacio de estados. Dado que las operaciones de ronda del AES son todas operaciones algebraicas, habran muchas posibilidades para la construcción de representaciones alternativas. Si una representación alternativa se basa en isomorfismos de álgebras, entonces la denominaremos como cifrado *isomorfo*.

5.3. Gran Sistema de Encriptación (BES)

Una representación del AES se consigue encajando el AES en un cifrado mayor llamado *Gran Sistema de Encriptación* (BES por sus siglas en ingles). El BES es definido como una forma de replicar la acción del AES usando simples operaciones algebraicas sobre $\mathbb{F}$. El BES opera sobre bloques de 128 bytes con claves de 128 bytes y tiene una muy simple estructura algebraica. Una ronda del BES consiste de la inversión de cada uno de estos 128 bytes y una transformación afín respecto a un espacio vectorial de dimensión 128 sobre $\mathbb{F}$.

5.3.1. Aplicación de Encaje del AES en el BES

Denotemos el álgebra de estados del AES por $\mathbb{A} = \mathbb{F}^{16}$ y el del BES por $\mathbb{B} = \mathbb{F}^{128}$. La aplicación para encajar el AES en el BES está basada en la aplicación vector conjugado $\phi : \mathbb{F} \to \mathbb{F}^8$, que a cada elemento de $\mathbb{F}$ le hace corresponder al vec-

tor formado por sus ocho conjugados. Esta aplicación ϕ es un homomorfismo inyectivo de anillos dado por

$$a \mapsto \left(a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3}, a^{2^4}, a^{2^5}, a^{2^6}, a^{2^7}\right)^T.$$

Esta definición se puede extender de manera obvia a una función encaje $\phi : \mathbb{A} \to \mathbb{B}$ definida por

$$(a_0, \ldots, a_{15})^T \mapsto (\phi(a_0), \ldots, \phi(a_{15}))^T,$$

que es también un homomorfismo inyectivo de anillos. Podemos por tanto usar ϕ para encajar un elemento del espacio de estados del AES $\mathbb{A}$ en el espacio de estados del BES $\mathbb{B}$ y definimos

$$\mathbb{B}_{\mathbb{A}} = \phi(\mathbb{A}) \subset \mathbb{B}$$

para ser la imagen encajada del espacio de estados del AES. Notemos que $\mathbb{B}_{\mathbb{A}}$ es un subanillo de $\mathbb{B}$ no así un subálgebra. A pesar e ello, $\mathbb{B}_{\mathbb{A}}$ contiene una base para $\mathbb{B}$ como espacio vectorial, y así $\mathbb{B}$ resulta ser la clausura de $\mathbb{B}_{\mathbb{A}}$. Como la aplicación inversa $\phi^{-1} : \mathbb{B}_{\mathbb{A}} \to \mathbb{A}$ está bien definida, el BES da lugar al siguiente diagrama conmutativo

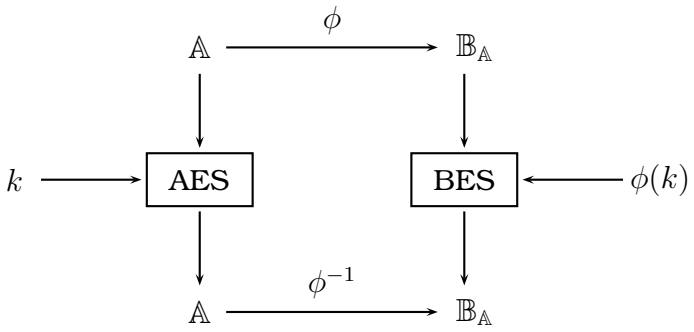

Figura 3. La relación entre el AES y el BES.

5.3.2. Estructura del BES

Las operaciones de inversión, **`ShiftRows`**, **`MixColumns`** y **`AddRoundKey`** en el AES con espacio de estados $\mathbb{A}$ son reemplazadas por las operaciones correspondientes obvias dentro del BES con espacio de estados $\mathbb{B}$. La adición de la constante 63 de la S-box en el AES es reemplazada por la operación correspondiente obvia dentro del BES con espacio de estados $\mathbb{B}$; pero acá se incorpora una clave de ronda revisada. La operación restante que aún no fue tomada en cuenta es la aplicación $\mathrm{GF}(2)$-lineal en la S-box de la operación **`SubBytes`**.

La aplicación $\mathrm{GF}(2)$-lineal está definida considerando el campo Rijndael $\mathbb{F}$ como un espacio vectorial de dimensión 8 sobre $\mathrm{GF}(2)$. Esto está implícitamente establecido en el AES por la aplicación natural de identificación $\psi : \mathbb{F} \to \mathrm{GF}(2)^8$. La operación $\mathrm{GF}(2)$-lineal del AES $\mathbb{F} \to \mathbb{F}$ por componente, queda entonces definida por $a \mapsto \psi^{-1}(\xi(\psi(a)))$, donde $\xi : \mathrm{GF}(2)^8 \to \mathrm{GF}(2)^8$ es la transformación

lineal dada en la subsección 4.4.6. La dependencia de las aplicaciones ψ y ψ^{-1} por parte del AES complica el análisis algebraico del mismo. La aplicación $\mathrm{GF}(2)$-lineal puede ser reemplazada por un polinomio linealizado sobre $\mathbb{F}$. Por lo tanto la aplicación $\mathrm{GF}(2)$-lineal estará dada por

$$a \mapsto \mathtt{05}a^{2^0} + \mathtt{09}a^{2^1} + \mathtt{F9}a^{2^2} + \mathtt{25}a^{2^3} + \mathtt{F4}a^{2^4} + \mathtt{01}a^{2^5} + \mathtt{B5}a^{2^6} + \mathtt{8F}a^{2^7}.$$

Esto significa que la operación $\mathrm{GF}(2)$-lineal en la S-box del AES puede ser definida al interior del BES por una matriz 8×8 sobre $\mathbb{F}$. Esta matriz replica la acción de la aplicación $\mathrm{GF}(2)$-lineal del AES sobre el primer byte de un conjunto de vectores conjugados y asegura que la propiedad de conjugación vectorial será preservada sobre los bytes restantes.

5.3.3. Función de Ronda del BES

La función de ronda del BES tiene la misma forma simple que la del AES. Consiste de una inversión seguida por una transformación afín. Supongamos que el estado al iniciarse la ronda del BES es $\mathbf{b} \in \mathbb{B}$ y que la clave de ronda del BES es $(\mathbf{k}_B)_i \in \mathbb{B}$, entonces la función de ronda del BES está dada por

$$\mathbf{b} \mapsto M_B \mathbf{b}^{(-1)} + (\mathbf{k}_B)_i,$$

donde M_B es una matriz de 128×128 sobre $\mathbb{F}$ que ejecuta la difusión lineal dentro del BES. La matriz de difusión M_B del BES y la matriz M de difusión del AES están relacionadas estrechamente y comparten propiedades algebraicas. M_B es concisa, tiene a $(x+1)^{15}$ como polinomio minimal y subespacios invariantes relacionados. Además, la función de ronda del AES puede ser expresada en términos del BES. Si el estado al principio de una ronda del AES es $\mathbf{w} \in \mathbb{A}$ y la clave de ronda es $\mathbf{k}_i^* \in \mathbb{A}$, entonces la función de ronda del AES está dada por

$$\mathbf{w} \mapsto M\mathbf{w}^{(-1)} + \mathbf{k}_i^* = \phi^{-1}(M_B(\phi(\mathbf{w})^{(-1)}) + \phi(\mathbf{k}_i^*)).$$

Así pues, el AES puede fácilmente ser definido en términos del BES.

Esta definición de la función de ronda del AES en términos del BES puede permitir que ciertas propiedades algebraicas del AES sean visibles directamente. Por ejemplo, las funciones componentes de salida en la operación de inversión están relacionadas por medio de transformaciones lineales, éstas proporcionan varios resultados acerca de la S-box y la función de ronda del AES.

5.3.4. Encaje o Inmersión del AES en el BES

El efecto de la aplicación encaje $\phi : \mathbb{A} \to \mathbb{B}$ sobre la función de encriptación del AES es el de inducir una función de encriptación inmersa $f_\phi : \mathbb{B}_\mathbb{A} \to \mathbb{B}_\mathbb{A}$. Esta función puede ser extendida de modo natural a una función $\overline{f}_\phi : \mathbb{B} \to \mathbb{B}$ y así el BES puede ser considerado naturalmente como la clausura de la inmersión vectorial conjugada del AES.

Que el BES pueda ser expresado usando simples operaciones algebraicas sobre un campo único $\mathbb{F}$, proporciona una percepción particularmente útil del cifrado. Usando el BES seremos capaces de obtener un sistema de ecuaciones cuadráticas multivariantes sobre $\mathrm{GF}(2^8)$ que describe una encriptación vía AES. Este sistema es más sencillo y reducido que aquél que se obtendria directamente desde el AES. La generación y posible solución de tales sistemas de ecuaciones multivariantes para el AES no es objeto del presente trabajo.

5.4. Otras Representaciones del AES

Ahora consideraremos otras representaciones del AES a las que llamaremos *cifrados duales*. A estas representaciones las clasificaremos de acuerdo a las propiedades de sus aplicaciones de representación.

5.4.1. Cifrados Isomorfos

Muchos de los cifrados duales son representaciones alternativas del AES donde las aplicaciones de los espacios de estados y los espacios de claves son isomorfismos de álgebras del álgebra de estados del AES. Los cifrados resultantes son por ende isomorfos al AES.

El campo finito $\mathrm{GF}(2^8)$ puede ser construido como una extensión de campo de cualquiera de sus subcampos, de aquí que pueden construirse representaciones isomorfas con base en la siguiente cadena de subcampos

$$\mathrm{GF}(2) \subset \mathrm{GF}(2^2) \subset \mathrm{GF}(2^4) \subset \mathrm{GF}(2^8).$$

Cada polinomio irreducible de grado d en $\mathrm{GF}(2^n)[x]$ puede ser usado para construir una extensión finita de grado d de $\mathrm{GF}(2^n)$ isomorfo a $\mathrm{GF}(2^{nd})$ (véase la sección 3.2).

EJEMPLO. El polinomio $x^4 + x + 1$ es irreducible en $\mathrm{GF}(2)[x]$ y denotamos una de sus raíces por ρ. Así tenemos

$$\mathbb{K} = \mathrm{GF}(2)(\rho) \cong \frac{\mathrm{GF}(2)[x]}{\langle x^4 + x + 1 \rangle} \cong \mathrm{GF}(2^4).$$

El polinomio $y^2 + y + \rho^3$ es irreducible en $\mathbb{K}[y]$. Si denotamos la raíz de $y^2 + y + \rho^3$ por ζ, entonces tendremos que

$$\mathbb{K}(\zeta) = (\mathrm{GF}(2)(\rho))(\zeta) \cong \frac{\mathbb{K}[y]}{\langle y^2 + y + \rho^3 \rangle} \cong \mathrm{GF}(2^8).$$

Podemos desde ya, representar cualquier elemento de $\mathbb{K}(\zeta)$ por $a_1\zeta + a_0$ para algunos $a_1, a_0 \in \mathbb{K}$. Como a_1 y a_0 pueden ser representados de modo natural por

caracteres hexadecimales, podemos interpretar al elemento $a_1\zeta + a_0$ como un par de caracteres hexadecimales (a_1, a_0). El elemento $\rho\zeta \in \mathbb{K}(\zeta)$ satisface

$$(\rho\zeta)^8 + (\rho\zeta)^4 + (\rho\zeta)^3 + (\rho\zeta) + 1 = 0,$$

así pues existe un isomorfismo de campos entre el campo de Rijndael $\mathbb{F} = \mathrm{GF}\,(2)(\theta)$, donde θ es la raíz de Rijndael, y $(\mathrm{GF}\,(2)(\rho))(\zeta)$ dado por $\theta \mapsto \rho\zeta$. En representación hexadecimal **02** $\mapsto$ (**2**, **0**). Otro ejemplo es dado por **05** $\mapsto$ (**4**, **7**) como

$$\theta^2 + 1 \mapsto (\rho\zeta)^2 + 1 = \rho^2(\zeta + \rho^3) + 1 = \rho^2\zeta + (\rho^5 + 1) = \rho^2\xi + (\rho^2 + \rho + 1).$$

Diferentes representaciones de $\mathrm{GF}\,(2^8)$ pueden ser construidas usando polinomios irreducibles de un grado apropiado en anillos de polinomios univariantes sobre subcampos de $\mathrm{GF}\,(2^8)$. El número de tales polinomios irreducibles se da en la siguiente tabla.

	Subcampos		
Grado	$\mathrm{GF}\,(2)$	$\mathrm{GF}\,(2^2)$	$\mathrm{GF}\,(2^4)$
2	1	6	120
4	3	60	-
8	30	-	-

Figura 4. Número de polinomios irreducibles sobre subcampos de GF(2^8).

En total existen $30 + (1 \cdot 60) + (3 \cdot 120) + (1 \cdot 6 \cdot 120) = 1170$ diferentes representaciones isomorfas del campo de Rijndael $\mathbb{F}$ basado sobre sus subcampos. Estas diferentes representaciones definen 1170 cifrados isomorfos al AES.

El automorfismo de Frobenius $z \mapsto z^2$ de $\mathbb{F}$ puede ser usado para obtener otros cifrados isomorfos. Este automorfismo de campos puede ser extendido de manera obvia para dar un isomorfismo de álgebras $\tau : \mathbb{F}^{16} \to \mathbb{F}^{16}$ del espacio de estados del AES. Para cualquier función $f : \mathbb{F}^{16} \to \mathbb{F}^{16}$ utilizada por el AES, podemos definir una función $f^{(2)} : \mathbb{F}^{16} \to \mathbb{F}^{16}$ por $x \mapsto \tau(f(\tau^{-1}(x)))$. Entonces podemos reemplazar f por $f^{(2)}$ y la clave k por $\tau(k)$ para obtener un nuevo cifrado $\mathcal{E}^{(2)}$. Este nuevo cifrado es una representación alternativa del AES y satisface

$$\mathcal{E}^{(2)}_{\tau(k)}(\tau(x)) = \tau(\mathcal{E}_k(x)).$$

Esta representación $\mathcal{E}^{(2)}$ es un cifrado isomorfo al que llamaremos *dual cuadrado*. Existen ocho automorfismos de Frobenius de $\mathbb{F}$. Si estas aplicaciones de Frobenius son combinadas con aplicaciones de subcampos, se podría construir 9360 cifrados isomorfos al AES.

Estas representaciones alternativas del AES están basadas en isomorfismos de campos de $\mathbb{F}$. Esto hace parecer improbable que ellas tengan algún interés criptoanalítico. No obstante, tales representaciones alternativas fueron propuestas en pos de mejorar la eficiencia de las implementaciones en hardware, mayormente con respecto a la transformación **SubBytes**.

5.4.2. Representaciones Regulares

La representación regular es una técnica matemática modelo muy poderosa para el estudio de un álgebra. Una *representación* de un $\mathbb{F}$-álgebra n-dimensional $\mathcal{A}$ es un homomorfismo de álgebras que va de $\mathcal{A}$ a un subálgebra del álgebra de matrices $\mathcal{M}_l(\mathbb{F})$ (ver ejemplo en capítulo 1). Una de estas representaciones estándar es la *representación regular*. Esta es el homomorfismo de álgebras $\nu : \mathcal{A} \to \mathcal{M}_n(\mathbb{K})$ que a cada $a \in \mathcal{A}$ le hace corresponder la matriz correspondiente a la transformación lineal $z \mapsto az$, donde z es un vector sobre $\mathbb{F}$ de longitud n.

EJEMPLO. Los números complejos forman un $\mathbb{R}$-álgebra 2-dimensional. El número complejo $x + iy$ puede ser identificado con su representación regular como una matriz, que está dada por

$$\nu(x + iy) = \begin{pmatrix} x & y \\ -y & x \end{pmatrix}.$$

El conjunto de todas éstas matrices forman un álgebra 2-dimensional sobre los números reales y puede ser identificada con el álgebra de los números complejos $\mathbb{C}$.

EJEMPLO. El campo de Rijndael $\mathbb{F}$ es un $\mathrm{GF}(2)$-álgebra 8-dimensional. La representación regular de $a \in \mathbb{F}$ es la matriz T_a de 8×8 vista en el ejemplo del capítulo 2. El conjunto de todas aquellas matrices $\{T_a | \, a \in \mathbb{F}$ forma un subálgebra 8-dimensional de $\mathcal{M}_8(\mathrm{GF}(2))$. La representación regular del campo de Rijndael $\mathbb{F}$ como un $\mathrm{GF}(2)$-álgebra es este subálgebra.

El espacio de estados del AES puede ser considerado como un $\mathrm{GF}(2)$-álgebra o como un $\mathbb{F}$-álgebra. El ejemplo anterior muestra que la representación regular del espacio de estados del AES como un $\mathrm{GF}(2)$-álgebra es un conjunto de matrices diagonales por bloques formando un subálgebra de $\mathcal{M}_{128}(\mathrm{GF}(2))$. Esta representación regular está dada por la aplicación

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{15} \end{pmatrix} \mapsto \begin{pmatrix} T_{a_0} & 0 & \cdots & 0 \\ 0 & T_{a_1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_{a_{15}} \end{pmatrix},$$

donde $(a_0, a_1, \ldots, a_{15})^T$ es interpretado como un vector sobre $\mathrm{GF}(2)$ de longitud 128. Entonces el proceso de encriptación AES puede ser definido en términos de operaciones matriciales convencionales.

- **Inversión**. Para la matriz diagonal por bloques A, ésta es la aplicación $A \mapsto A^{-1} = A^{254}$. Es la inversión matricial si A es inversible.
- **Difusión lineal aumentada**. Para la matriz diagonal por bloques A, existen matrices diagonales por bloques D_i y matrices de permutación P_i para

$0 \leq i \leq 31$ tales que esta transformación lineal puede ser definida por

$$A \mapsto \sum_{i=0}^{31} D_i P_i A P_i^T.$$

- **Adición de la clave de ronda**. Para la matriz diagonal por bloques A y la matriz de clave de ronda K, ésta es la aplicación $A \mapsto A + K$.

El espacio de estados del AES es también un $\mathbb{F}$-álgebra, con representación regular dada por el homomorfismo de álgebras $\mathbb{F}^{16} \to \mathcal{M}_{16}(\mathbb{F})$ definida por

$$\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{15} \end{pmatrix} \mapsto \begin{pmatrix} x_0 & 0 & \cdots & 0 \\ 0 & x_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{15} \end{pmatrix}.$$

Así la representación regular del espacio de estados del AES $\mathbb{F}^{16}$ es $\mathcal{D}_{16}(\mathbb{F})$, el $\mathbb{F}$-álgebra de matrices de 16×16. Similarmente, el álgebra $\mathcal{D}_{128}(\mathbb{F})$ de matrices diagonales de 128×128 es la representación regular del espacio de estados BES $\mathbb{F}^{128}$. Esto da una inmersión de un elemento del AES subconjunto del BES definido por

$$\begin{pmatrix} x_0^{2^0} \\ x_0^{2^1} \\ \vdots \\ x_0^{2^7} \\ x_1^{2^0} \\ \vdots \\ x_{15}^{2^7} \end{pmatrix} \mapsto \begin{pmatrix} x_0^{2^0} & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & x_0^{2^1} & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & x_0^{2^7} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & x_1^{2^0} & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & x_{15}^{2^7} \end{pmatrix}.$$

La representación regular del AES como subconjunto del BES es el subanillo de las matrices diagonales donde los octetos forman conjuntos de conjugados. El proceso de encriptación del BES, y por tanto el proceso de encriptación del AES, pueden ser definidos en términos de la matriz diagonal B de un espacio vectorial de estados del BES.

- **Inversión**. $B \mapsto B^{(-1)} = B^{254}$.
- **Difusión lineal**. $B \mapsto \sum_{i=0}^{31} D_i P_i B P_i^T$, donde las D_i son matrices diagonales y las P_i son matrices de permutación.
- **Adición de la Subclave**. $B \mapsto B + K$ donde K es la representación regular de la clave de ronda.

5.4.3. Representaciones Logarítmicas

El AES está especificado a través del uso de una representación polinomial para los elementos de $\mathbb{F}$. Pero también podemos representar un elemento de $\mathbb{F}$ como un elemento de $\overline{\mathbb{Z}}_{255}$ empleando el logaritmo discreto. Así podemos dar la *representación logarítmica* de un elemento del espacio de estados del AES como un elemento del conjunto $(\overline{\mathbb{Z}}_{255})^{16}$. Representaciones alternativas basadas en la representación logarítmica de $\mathbb{F}$ son llamadas cifrados *log-dual*. Existen 128 elementos primitivos en $\mathbb{F}$ dando como resultado 128 cifrados log-dual diferentes para el AES.

Una operación multiplicativa es fácilmente formulada en la representación logarítmica, y una operación aditiva puede ser definida en términos del Logaritmo de Zech. Podemos representar directamente un elemento de $\mathbb{F}$ como un elemento de $\overline{\mathbb{Z}}_{255}$ usando el logaritmo de Zech y ésto da una *representación logaritmo de Zech* del espacio de estados del AES como un elemento del conjunto $(\overline{\mathbb{Z}}_{255})^{16}$. Esto conduciría a una más resumida descripción del AES que la representación logarítmica convencional.

5.5. Propiedades Inherentes a la Teoría de Grupos

Fundamentalmente, un cifrado por bloques provee una breve descripción para un conjunto indexado de permutaciones sobre el espacio de estados. En consecuencia podriamos intentar obtener mayor comprensión con respecto a la estructura de un cifrado por bloques considerando sus transformaciones como elementos de algún grupo de permutaciones. De igual manera, las funciones que constituyen la ronda de un cifrado por bloques iterado también forman conjuntos de permutaciones y pueden ser analizadas desde la misma perspectiva. El propósito principal de esta sección es la consideración de un análisis sobre las funciones involucradas en la encriptación y la función de ronda del AES, como permutaciones actuando sobre el espacio de estados.

5.5.1. Cifrados Iterados por Bloques

Suponga que un cifrado por bloques tiene un espacio de estados $\mathcal{X}$ y un espacio de claves $\mathcal{K}$. Para una clave dada $k \in \mathcal{K}$, la encriptación vía el cifrado por bloques es una permutación $\varepsilon : \mathcal{X} \to \mathcal{X}$. El conjunto $\mathcal{E} = \{\varepsilon_k | \, k \in \mathcal{K}\}$ de todas las posibles encriptaciones del cifrado por bloques es un subconjunto de $S_{\mathcal{X}}$, el grupo de todas las permutaciones de elementos del espacio de estados. El grupo $\mathcal{G} = \{\varepsilon_k | k \in \mathcal{K}\}$ generado por el conjunto de las permutaciones del cifrado es conocido como el *grupo generado por el cifrado*. Si $\mathcal{G} = \mathcal{E}$, es decir que el conjunto de permutaciones $\{\varepsilon_k | \, k \in \mathcal{K}\}$ forman un grupo, entonces decimos que el cifrado es un grupo. Como $\mathcal{G}$ es un grupo finito, el cifrado es un grupo si y sólo si el conjunto $\mathcal{E}$ es cerrado bajo la operación de composición. Para un cifrado tal, la múltiple encriptación no ofrece seguridad extra que la dada por

una sóla encriptación.

Más generalmente, ciertas propiedades del grupo $\mathcal{G}$ generado por un cifrado son de interés criptográfico ya que han sido propuestos ataques contra cifrados que no satisfacen algunas de estas propiedades. Sin embargo propiedades favorables de grupo no constituyen una garantía de un cifrado fuerte y resistente.

El cálculo del grupo $\mathcal{G}$ generado por un cifrado por bloques es frecuentemente dificultoso. Sea v_{k_i} la función de ronda del cifrado bajo la subclave $k_i \in \mathcal{K}_s$ donde $\mathcal{K}_s$ es el espacio de las subclaves de ronda. Las funciones de ronda v_{k_i} que estructuran una encriptación ε_k son también permutaciones del espacio de estados $\mathcal{X}$ y resulta que a menudo es más fácil calcular los varios grupos generados por estas permutaciones. Supongamos que tenemos un cifrado por bloques de r-rondas con una función de programación de claves $KS : \mathcal{K} \to (\mathcal{K}_s)^r$, así que cualquier clave $k \in \mathcal{K}$ dará lugar a r subclaves en $\mathcal{K}_s$. Las permutaciones de la función de ronda sugieren naturalmente los tres siguientes grupos de relevancia para el cifrado por bloques:

$$\begin{aligned}
\mathcal{R} &= \langle v_k |\, k \in \mathcal{K}_s \rangle, \\
\mathcal{P} &= \langle v_{k_r} \cdots v_{k_2} v_{k_1} |\, k_i \in \mathcal{K}_s \rangle, \\
\mathcal{G} &= \langle v_{k_r} \cdots v_{k_2} v_{k_1} |\, KS(k) = (k_1, \ldots, k_r) \rangle = \langle \varepsilon_k |\, k \in \mathcal{K} \rangle.
\end{aligned}$$

Así $\mathcal{R}$ es el grupo generado por las funciones de ronda y $\mathcal{P}$ es el grupo generado por una arbitraria composición de r funciones de ronda. El grupo $\mathcal{G}$ generado por el cifrado puede también ser considerado como el grupo generado por cualquier composición de r funciones de ronda permitidas por el esquema de programación de claves. La relación entre estos grupos es que $\mathcal{G}$ es un subgrupo de $\mathcal{P}$, y $\mathcal{P}$ es un subgrupo normal de $\mathcal{R}$ ($\mathcal{G} < \mathcal{P} \lhd \mathcal{R}$). Así el grupo generado por las funciones de ronda acota superiormente al grupo generado por el cifrado.

EJEMPLO. Las propiedades de los grupos generados por el DES han sido extensamente estudiadas. Inicialmente, se observó que las estructuras de ciclo de ciertas permutaciones podían ser usadas para obtener una cota inferior para el orden del grupo $\mathcal{G}$ generado por el DES. Subsecuentemente, las estructuras de ciclo fueron analizadas en profundidad y utilizadas para mostrar que $|\mathcal{G}| > 2^{56}$, de donde se concluye que el DES no es un grupo.

La función de ronda del DES bajo cualquier clave, y de aquí cualquier encriptación vía DES, es un permutación par. Además, el grupo $\mathcal{R}$ generado por las funciones de ronda del DES es el grupo alternado $A_{2^{64}}$ sobre el espacio de estados del DES, que es un grande, simple, primitivo y altamente transitivo grupo. Así el grupo $\mathcal{P} \lhd \mathcal{R}$ generado por la composición de cualquier número fijo de funciones de ronda es también el grupo alternado $A_{2^{64}}$. Se sigue que el grupo $\mathcal{G}$ generado por el DES es un subgrupo del grupo alternado $A_{2^{64}}$, aunque poco más se sabe sobre esta estructura.

5.5.2. Estructuras de Ciclo

Ahora trataremos las estructuras de ciclo de las diferentes operaciones en la función de ronda del AES cuando son consideradas como permutaciones del espacio de estados.

Primero consideremos la permutación π_{ij} del espacio de estados del AES de tamaño 2^{128} dada por la aplicación de la S-box del AES al byte $\mathcal{S}_{ij}$, mientras los restantes quince bytes se mantienen fijos. La acción de la operación **SubBytes** sobre el espacio de estados del AES está dada por la permutación $\pi_{00} \cdots \pi_{33}$. Esta permutación es el producto a 16 componentes de permutaciones que tienen la misma estructura de ciclo. Por tanto la operación **SubBytes** es una permutación par del espacio de estados del AES.

Ahora consideremos las operaciones **ShiftRows** y **MixColumns**. Dijimos al principio del presente capítulo que estas operaciones pueden ser definidas en términos de la aplicación de la matriz de permutación $\widehat{R}$ a cuatro bytes del espacio de estados del AES, mientras los restantes 12 bytes quedan fijos, todo ésto con respecto a alguna base. Denotamos la permutación del espacio de estados del AES dada por una aplicación de la matriz de permutación $\widehat{R}$, con respecto a una base apropiada, para una fila i por $\pi^{(i\cdot)}$ y para una columna j por $\pi^{(\cdot j)}$. Entonces la operación **ShiftRows** y la operación **MixColumns** estarán dadas respectivamente por las permutaciones

$$\left(\pi^{(1\cdot)}\right)\left(\pi^{(2\cdot)}\right)^2\left(\pi^{(3\cdot)}\right)^3 \text{ y } \left(\pi^{(\cdot 0)}\right)\left(\pi^{(\cdot 1)}\right)\left(\pi^{(\cdot 2)}\right)\left(\pi^{(\cdot 3)}\right).$$

Estos son productos de permutaciones con la misma estructura de ciclo a 6 y 4 componentes, por lo que ambas operaciones son permutaciones pares del espacio de estados del AES.

Finalmente consideremos la operación **AddRoundKey**. Está claro, que esta operación es el producto de 2^{127} transposiciones para una clave de ronda no nula y la transformación identidad para la clave de ronda nula, por lo que la operación **AddRoundKey** es una permutación par.

Todas las operaciones usadas por las funciones de ronda del AES son por lo tanto permutaciones pares del espacio de estados del AES, lo que prueba el siguiente teorema.

TEOREMA 1. La función de ronda del AES es una permutación par.

La estructura de ciclos de cada componente de la función de ronda del AES puede ser en gran parte deducida a partir de la tabla de la Figura 1 y están dadas en la tabla de la Figura 5. Nótese que la permutación S-box del AES sobre los 2^8 elementos de $\mathbb{F}$ tiene cinco ciclos disjuntos de longitudes 87, 81, 59, 27 y 2, por lo que es una permutación impar. Un análisis detallado de las varias permutaciones generadas por las funciones de ronda del AES son dadas por el Teorema 2 a continuación.

TEOREMA 2. El grupo $\mathcal{R}$ generado por las funciones de ronda del AES es el grupo alternado $A_{2^{128}}$.

Esto significa que el grupo $\mathcal{P}$ para el AES es también el grupo alternado; lo cual implica que "desde el punto de vista algebraico considerables debilidades (del AES) pueden ser excluidas". Nótese que el grupo $\mathcal{G}$ generado por el AES no es sólo una simple composición de funciones de ronda, porque se tiene una operación **`AddRoundKey`** inicial, y no se incluye la operación **`MixColumns`** en la ronda final. Sin embargo, las operaciones de ronda resultantes son permutaciones pares, y así tenemos además que $\mathcal{G} < A_{2^{128}}$.

	Número de ciclos			Paridad de la
	fijos	orden 2	orden 4	permutación
Inversión	2^{16}	$2^{-1}(2^{128}-2^{16})$	0	Par
GF(2)-lineal	2^{48}	$2^{-1}(2^{96}-2^{48})$	$2^{-2}(2^{128}-2^{96})$	Par
Constante de la S-box	0	2^{127}	0	Par
`ShiftRows`	2^{64}	$2^{-1}(2^{96}-2^{64})$	$2^{-2}(2^{128}-2^{96})$	Par
`MixColumns`	2^{32}	$2^{-1}(2^{64}-2^{32})$	$2^{-2}(2^{128}-2^{64})$	Par
`AddRoundKey`	0	2^{127}	0	Par
(Clave de ronda 0)	(2^{128})	(0)	(0)	(Par)

Figura 5. La estructura de ciclo de los diferentes componentes del AES.

5.5.3. Grupo de Difusión de Bytes

Supongamos que G es un grupo de permutaciones de los 16 bytes del arreglo de estados, así $G < S_{16}$. Cada elemento $g \in G$ sirve para definir una permutación $g_{\mathbb{A}} \in S_{\mathbb{A}}$ del espacio de estados del AES $\mathbb{A} = \mathbb{F}^{16}$, dicha definición es presentada en siguiente tabla

Aplicación	Definición
$g_{\mathbb{A}}$	$(\mathcal{S}_{00},\ldots,\mathcal{S}_{33})^T \mapsto (\mathcal{S}_{g(00)},\ldots,\mathcal{S}_{g(33)})^T$
$(gg')_{\mathbb{A}}$	$(\mathcal{S}_{00},\ldots,\mathcal{S}_{33})^T \mapsto (\mathcal{S}_{g(g'(00))},\ldots,\mathcal{S}_{g(g'(33))})^T$
$g_{\mathbb{A}} + g'_{\mathbb{A}}$	$(\mathcal{S}_{00},\ldots,\mathcal{S}_{33})^T \mapsto (\mathcal{S}_{g(00)}+\mathcal{S}_{g'(00)},\ldots,\mathcal{S}_{g(33)}+\mathcal{S}_{g'(33)})^T$
$\lambda \cdot g_{\mathbb{A}}$	$(\mathcal{S}_{00},\ldots,\mathcal{S}_{33})^T \mapsto (\lambda\mathcal{S}_{00},\ldots,\lambda\mathcal{S}_{33})^T$

Figura 6. Definiciones de algunas aplicaciones sobre el espacio de estados $\mathbb{A}$ del AES.

Además, tales aplicaciones pueden ser extendidas a las aplicaciones $(gg')_{\mathbb{A}}$, $g_{\mathbb{A}} + g'_{\mathbb{A}}$ y $\lambda \cdot g_{\mathbb{A}}$ (con $g, g' \in G$ y $\lambda \in \mathbb{F}$) del espacio de estados $\mathbb{A}$ del AES (véase la Figura 6). Así, cualquier suma formal

$$\sum_{g\in G} \lambda_g \cdot g \qquad\qquad (\text{con } \lambda_g \in \mathbb{F})$$

puede ser empleada para definir una aplicación $\left(\sum_{g\in G}\lambda_g\cdot g\right)_{\mathbb{A}}$ del espacio de estados $\mathbb{A}$ del AES dada por

$$(\mathcal{S}_{00},\ldots,\mathcal{S}_{33})^T \mapsto \sum_{g\in G}\lambda_g\cdot\left(\mathcal{S}_{g(00)},\ldots,\mathcal{S}_{g(33)}\right)^T.$$

El conjunto de todas las sumas formales de elementos de G bajo las reglas obvias de adición y multiplicación forman un álgebra, conocido como el *grupo-álgebra* de G sobre $\mathbb{F}$, y es denotado por $\mathbb{F}[G]$.

Ahora trataremos las permutaciones de los 16 bytes del arreglo de estados, necesarias para definir las operaciones **`ShiftRows`** y **`MixColumns`**. Ambas operaciones pueden ser definidas en términos de una permutación específica de los 16 bytes del arreglo. Luego cada operación puede ser definida en términos de un sólo elemento del grupo-álgebra $\mathbb{F}[\mathcal{S}_{16}]$. La permutación

$$\varrho = (\mathcal{S}_{10}\mathcal{S}_{13}\mathcal{S}_{12}\mathcal{S}_{11})(\mathcal{S}_{20}\mathcal{S}_{22})(\mathcal{S}_{21}\mathcal{S}_{23})(\mathcal{S}_{30}\mathcal{S}_{31}\mathcal{S}_{32}\mathcal{S}_{33})$$

es la permutación de bytes definida por la operación **`ShiftRows`**. Así, el elemento $\Delta = 1\cdot\varrho \in \mathbb{F}[\mathcal{S}_{16}]$ del grupo-álgebra genera la aplicación $\Delta_{\mathbb{A}}$, que es la operación **`ShiftRows`**. De manera similar, una rotación simultánea en una posición de todas las columnas en el arreglo de estados define la permutación

$$\varsigma = (\mathcal{S}_{00}\mathcal{S}_{10}\mathcal{S}_{20}\mathcal{S}_{30})(\mathcal{S}_{01}\mathcal{S}_{11}\mathcal{S}_{21}\mathcal{S}_{31})(\mathcal{S}_{02}\mathcal{S}_{12}\mathcal{S}_{22}\mathcal{S}_{32})(\mathcal{S}_{03}\mathcal{S}_{13}\mathcal{S}_{23}\mathcal{S}_{33}).$$

Entonces podemos definir el elemento del grupo-álgebra $\Gamma \in \mathbb{F}[\mathcal{S}_{16}]$ por

$$\Gamma = \theta\cdot e + (\theta+1)\cdot\varsigma + 1\cdot\varsigma^2 + 1\cdot\varsigma^3,$$

donde e denota el elemento identidad de $\mathcal{S}_{16}$. La operación **`MixColumns`** es entonces dada por la aplicación $\Gamma_{\mathbb{A}}$. La difusión de bytes de la función de ronda del AES está dada por la operación **`ShiftRows`** seguida por la operación **`MixColumns`**. Así $\Lambda_{\mathbb{A}} = (\Gamma\Delta)_{\mathbb{A}}$ especifica la difusión, donde el elemento del grupo-álgebra $\Lambda \in \mathbb{F}[\mathcal{S}_{16}]$ está dado por

$$\Lambda = \Gamma\Delta = \theta\cdot\varrho + (\theta+1)\cdot\varsigma\varrho + 1\cdot\varsigma^2\varrho + 1\cdot\varsigma^3\varrho.$$

El entreverado de los bytes requerido por la estrategia del rastro amplio es por tanto dado por los dos elementos Δ y Γ del grupo-álgebra $\mathbb{F}[\mathcal{S}_{16}]$, que depende enteramente de las dos permutaciones ϱ y ς de $\mathbb{F}[\mathcal{S}_{16}]$. Si definimos el subgrupo $H = \langle\varrho,\varsigma\rangle < \mathcal{S}_{16}$, entonces las operaciones **`ShiftRows`** y **`MixColumns`** estarán dadas por elementos del más pequeño grupo-álgebra $\mathbb{F}[H]$. La difusión de bytes requerida para el AES por la estrategia del rastro amplio puede en tanto ser definida en términos del grupo-álgebra $\mathbb{F}[H]$ y así, llamaremos a H el *grupo de difusión de bytes* del AES.

Ahora consideremos este grupo de difusión de bytes $H = \langle\varrho,\varsigma\rangle$. Primero nótese que ϱ y ς son permutaciones pares, por lo que $H < A_{16}$. Con el propósito de

tratar el grupo de difusión H definimos la permutación $\varphi = \varsigma\varrho\varsigma^{-1}\varrho^{-1} \in H$, así tenemos que

$$\varphi = (\mathcal{S}_{00}\mathcal{S}_{03}\mathcal{S}_{02}\mathcal{S}_{01})(\mathcal{S}_{10}\mathcal{S}_{13}\mathcal{S}_{12}\mathcal{S}_{11})(\mathcal{S}_{20}\mathcal{S}_{23}\mathcal{S}_{22}\mathcal{S}_{21})(\mathcal{S}_{30}\mathcal{S}_{33}\mathcal{S}_{32}\mathcal{S}_{31}).$$

La permutación φ corresponde a una rotación simultanea de todas las filas en una posición. Nótese además que φ conmuta con ϱ y ς, por lo que $\Lambda(1 \cdot \varphi) = (1 \cdot \varphi)\Lambda$. Esta observación muestra que ϕ es una permutación de los bytes del arreglo de estados que puede ser utilizada para definir una función de ronda auto-dual para el AES.

Con la intención de describir el grupo de difusión de bytes H, definimos el subgrupo $H_0 = \langle\varsigma\rangle < H$ y el subgrupo normal $N = \langle\varrho, \varphi\rangle \lhd H$. Los elementos del grupo H_0 permutan las filas del arreglo de estados, mientras que los elementos del subgrupo normal N actúan íntegramente en cada fila. El subgrupo H_0 es isomorfo al grupo cíclico C_4 de cuatro elementos, y el subgrupo normal N es isomorfo a $C_4 \times C_4$ y tiene 16 elementos. Además se puede mostrar que cualquier elemento de H puede ser expresado únicamente como el producto de un elemento del subgrupo normal N y un elemento de su subgrupo H_0. Se sigue que el grupo de difusión de bytes H del AES es el *producto semidirecto* de N por H_0 y que por lo tanto tiene orden 64.

Esta formulación del grupo difusión de bytes H muestra que su acción sobre el arreglo de estados consta de dos distintas partes. Puede dividirse en la acción del subgrupo normal N actuando íntegramente en cada fila y la acción del subgrupo H_0 que permuta las filas. Tal propiedad significa que H es un grupo imprimitivo y que cada fila del arreglo de estados es un bloque de imprimitividad. Más generalmente, hemos mostrado que cualquier difusión de bytes generada por la parte de la difusión lineal (**`ShiftRows`** y **`MixColumns`**) de la función de ronda del AES, puede ser dada por una suma formal de elementos de un pequeño grupo de difusión de bytes H de tamaño 64. Este grupo de difusión de bytes H del AES es pequeño y estructurado en comparación a un posible grupo de difusión de bytes de $\mathcal{S}_{16}$.

Tal análisis se extiende al espacio de estados $\mathbb{F}^{128}$ del BES. Un grupo de difusión de bytes para el BES de tamaño 512 es así obtenido y resulta que es un subgrupo de A_{128}. Este grupo de difusión es isomorfo al producto directo del grupo cíclico con 8 elementos y el grupo difusión H del AES.

5.5.4. Propiedades Geométricas

Las operaciones usadas en el AES pueden ser visualizadas como transformaciones geométricas. La inversión en un campo finito es una transformación geométrica en una geometría proyectiva, mientras que la difusión aumentada y la adición de clave de ronda forman una transformación afín de un espacio vectorial. Esto nos conduce a la observación de algunas propiedades geométricas del AES. No obstante nuestro enfoque se centrará en la recta proyectiva $\overline{\mathbb{F}}$ del campo de Rijndael $\mathbb{F}$, que se define a continuación.

DEFINICIÓN. Si $\mathbb{F}$ es el campo de Rijndael y $\mathbb{F}^2$ es el espacio vectorial de dimensión 2 sobre $\mathbb{F}$, la *recta proyectiva* $\overline{\mathbb{F}}$ del campo de Rijndael es el conjunto de subespacios unidimensionales del espacio vectorial $\mathbb{F}^2$ bajo la acción del grupo de transformaciones lineales invertibles de $\mathbb{F}^2$.

Los puntos de la recta proyectiva $\overline{\mathbb{F}}$ son todos los subespacios unidimensionales de $\mathbb{F}^2$, esto es

$$\overline{\mathbb{F}} = \{\langle(1, z)\rangle |\ z \in \mathbb{F}\} \cup \{\langle(0, 1)\rangle\}.$$

El punto proyectivo $\langle(0, 1)\rangle$ es conocido como el *punto al infinito*, de esta manera podemos considerar la recta proyectiva $\overline{\mathbb{F}}$ como $\mathbb{F} \cup \{\infty\}$. El grupo de transformaciones sobre $\overline{\mathbb{F}}$ es conocido como el *Grupo Lineal General Proyectivo* $PGL(2, \mathbb{F})$, un grupo de orden $2^8(2^{16} - 1)$. Además, $PGL(2, \mathbb{F})$ es un grupo fuertemente 3-transitivo, lo cual significa que la acción de un elemento sobre tres puntos proyectivos únicamente identifica a ese elemento.

El potencial de un enfoque geométrico se evidencia en el siguiente ejemplo, por medio del análisis de dos simples cifrados por bloques con espacios de estado $\mathbb{F}$ y $\overline{\mathbb{F}}$ respectivamente, veamos.

EJEMPLO. Dos cifrados $\mathcal{C}$ y $\overline{\mathcal{C}}$ con espacios de estado $\mathbb{F}$ y $\overline{\mathbb{F}}$ respectivamente son definidos abajo (con la convencional interpretación para ∞).

Cifrado	Espacio de estado	Clave de ronda	Función de ronda	Definición
$\mathcal{C}$	$\mathbb{F}$	$k \in \mathbb{F}$	$f_k : \mathbb{F} \to \mathbb{F}$	$x \mapsto x^{-1} + k$
$\overline{\mathcal{C}}$	$\overline{\mathbb{F}}$	$k \in \mathbb{F}$	$\overline{f_k} : \overline{\mathbb{F}} \to \overline{\mathbb{F}}$	$x \mapsto \frac{1}{x} + k$

Las funciones de ronda f_k y $\overline{f_k}$ de los dos cifrados coinciden sobre $\mathbb{F}$ a menos que se defina una 0-inversión. Así los cifrados $\mathcal{C}$ y $\overline{\mathcal{C}}$ transforman un texto-franco dado en el mismo texto-cifrado para la mayoría de los textos-francos.

Ahora consideremos los dos grupos

$$\mathcal{R} = \langle f_k |\ k \in \mathbb{F}\rangle \cong \text{Sym}(\mathbb{F}) \text{ y } \overline{\mathcal{R}} = \langle \overline{f_k} |\ k \in \mathbb{F}\rangle \cong PGL(2, \mathbb{F})$$

generados por las funciones de ronda para los dos cifrados (véase la subsección 5.5.1). Vemos que las funciones de ronda de $\mathcal{C}$ generan el grupo simétrico sobre $\mathbb{F}$, por lo tanto se requeriría muchos pares de (texto-franco,texto-cifrado) para determinar la transformación cifradora. En contraste, las funciones de ronda de $\overline{\mathcal{C}}$ que generan a $PGL(2, \mathbb{F})$, necesitan tan sólo tres pares de (texto-franco,texto-cifrado) para determinar la transformación cifradora.

Sin embargo, $\mathcal{C}$ y $\overline{\mathcal{C}}$ encriptan la mayoría de los textos-francos de la misma manera. Así, la transformación cifradora total de $\mathcal{C}$ puede también ser determinada, con tres pares de (texto-franco,texto-cifrado) con alta probabilidad.

El análisis dado en este último ejemplo es esencialmente una versión del ataque de interpolación desde el punto de vista de la teoría de grupos, para este

tipo de cifrados por bloques. Además, nos muestra como un cambio prácticamente insignificante a la definición de la acción de grupo considerada, puede producir un grupo muy diferente. Más aún, en este caso el grupo $\overline{\mathcal{R}}$ generado por la acción de grupo modificada es por mucho un indicador más preciso de la resistencia del cifrado a un ataque algebraico que el grupo $\mathcal{R}$ generado por la acción de grupo inalterada.

Parte III

Rijndael-GF

Capítulo 6

Análisis de la Propagación en Campos de Galois

Se requerirá una consulta adicional de [1] con el propósito de fijar ideas y facilitar la comprensión de los tópicos cubiertos en éste capítulo. Se invita al lector aleccionarse a través del mencionado material.

En la especificación del algoritmo Rijndael, hemos hecho un uso extensivo de operaciones en un campo finito, donde los bytes del estado y de la clave representan elementos de GF (2^8). Todavía, como para la mayoría de los cifrados por bloques, Rijndael opera sobre bloques de texto-franco, bloques de texto-cifrado y claves que son cadenas de bits. Al margen de algunas excepciones tales como los ataques de interpolación, el criptoanálisis está también conducido hacia el nivel de los bits. Por ejemplo, el criptoanálisis lineal, explota la alta correlación entre combinaciones lineales de bits del estado, en diferentes fases del proceso de encriptación. El criptoanálisis diferencial, aprovecha altas probabilidades de propagación entre diferencias de bits en el estado, en diferentes fases de la encriptación.

Demostraremos más adelante cómo Rijndael puede ser especificado completamente con operaciones en GF (2^8). Cómo los elementos de GF (2^8) que están representados en bytes, pueden ser vistos como un detalle de la especificación. El direccionamiento de este cuestión de representación en las especificaciones, es importante para diferentes implementaciones de Rijndael que lo hagan interoperable; pero no más que por ejemplo en el sentido del ordenamiento de los bits al interior de los bytes, o la manera en la que los bytes propios de los bloques de texto-franco y texto-cifrado son aplicados suryectivamente sobre los bytes del estado. Puede bien ser posible que tomando una representacion diferente del campo finito, ésto nos conduzca a más eficientes implementaciones.

Podemos hacer una abstracción de la representación de los elementos de GF (2^8) y considerar un cifrado por bloques como aquél que opera sobre cadenas de elementos de GF (2^8). Llamamos a ésta generalización Rijndael-GF. Rijndael puede ser considerado como un caso particular del Rijndael-GF, donde ya se ha especificado una representación de los elementos. En principio, esto puede

ser aplicado a la mayoría de los cifrados por bloques. Cada cifrado por bloques para el cual el tamaño del bloque y la longitud de la clave son multiplos de n pueden de inicio ser generalizados para operar sobre cadenas de elementos de GF(2^n). Sin embargo, a diferencia del cifrado Rijndael, la especificación de estos cifrados generalizados puede convertirse en un proceso muy complicado.

Intuitivamente, parece obvio que si Rijndael tiene una debilidad criptográfica, esta es inherente al Rijndael-GF y cualquier instancia en él, cualquiera sea la representación de los elementos de GF(2^8). En el análisis de la propagación para funciones Booleanas, se trabaja en el ámbito de los bits y además es preciso escoger una representación específica para poder así estudiar las propiedades de propagación. En éste capítulo, demostraremos cómo llevar el análisis de la propagación lineal y diferencial a nivel de los elementos de GF(2^n), sin tener que recurrir a representaciones.

Nos concentraremos en funciones sobre campos de característica 2, sin embargo, algunas de las propiedades y teoremas son válidos para cualquier campo finito. En casos en los que se trate el caso general, los campos de característica 2 son apenas un caso particular. Empezaremos por describir las propiedades de propagación de la diferencia y correlación de funciones sobre GF(2^n) haciendo hincapié en funciones que son lineales, lo cual será generalizado a funciones sobre $(\text{GF}(2^n))^l$.

6.1. Funciones sobre $\text{GF}(2^n)$

En esta sección estudiaremos la propagación diferencial y las propiedades de correlación de las funciones definidas sobre GF(2^n):

$$f : \text{GF}(2^n) \to \text{GF}(2^n) : a \mapsto b = f(a).$$

Todas las funciones sobre GF(2^n) pueden ser expresadas como un polinomio sobre GF(2^n) de grado a lo mas $2^n - 1$:

$$f(a) = \sum_{i=0}^{2^n-1} c_i a^i.$$

Dada una tabla donde se especifiquen los valores que $f(a)$ toma para los 2^n valores diferentes de a, los 2^n coeficientes de este polinomio pueden ser encontrados aplicando la interpolación de Lagrange. Por otro lado, dada una expresión polinomial, la tabla puede ser especificada por evaluación del polinomio para todos los valores de a.

6.1.1. Propagación de la Diferencia

Como para funciones Booleanas, tenemos que la probabilidad de propagación de la diferencia sobre f está dada por:

$$\text{Prob}^f(a', b') = 2^{-n} \sum_a \delta(b' + f(a + a') + f(a)),$$

donde $+$ denota la adición en $\mathrm{GF}(2^n)$. Una diferencia a' se propaga a una desviación b' a través de de f si, y sólo si:

$$f(a) + f(a + a') = b'.$$

La probabilidad de la propagación de la diferencia $\mathrm{Prob}^f(a', b')$ es igual al número total de valores a que satisfacen esta ecuación, dividido por 2^n. Empleando la expresión polinomial para f, tenemos

$$\sum_i c_i a^i + \sum_i c_i (a + a')^i + b' = 0.$$

Esta es una ecuación polinomial en a. Para ciertos casos especiales, el número de soluciones de estos polinomios puede ser determinado analíticamente, proporcionando cotas demostrables para la probabilidad de la propagación de la diferencia.

6.1.2. Correlación

Para funciones Booleanas, la correlación se define entre paridades. Las paridades son combinaciones lineales de bits, que son determinadas por un patrón de selección. Para una función sobre $\mathrm{GF}(2^n)$, los bits no pueden distinguirse de manera individual sin adoptar una representación, y por eso, hablar sobre paridades no tiene sentido. Una paridad es una función que aplica $\mathrm{GF}(2^n)$ en $\mathrm{GF}(2^n)$, la cual es lineal sobre $\mathrm{GF}(2)$. En $\mathrm{GF}(2^n)$ podemos encontrar funciones con las mismas propiedades. Para este propósito, utilizaremos la función *traza* sobre $\mathrm{GF}(p^n)$ con respecto a $\mathrm{GF}(2)$

De la definición de la función traza en la sección 3.4 se sigue que dado x un elemento de $\mathrm{GF}(p^n)$. La traza de x con respecto a $\mathrm{GF}(p)$ está dada por

$$\mathrm{Tr}_{\mathrm{GF}(p^n)/\mathrm{GF}(p)}(x) = x + x^p + x^{p^2} + x^{p^3} + \cdots + x^{p^{n-1}}.$$

Por simplicidad durante este capítulo y dado que queda claro el contexto, nos referiremos a la traza sólo como Tr.

La traza es lineal sobre $\mathrm{GF}(p)$, es decir

$$\forall x, y \in \mathrm{GF}(p^n) : \mathrm{Tr}(x + y) = \mathrm{Tr}(x) + \mathrm{Tr}(y)$$
$$\forall a \in \mathrm{GF}(p), \forall x \in \mathrm{GF}(p^n) : \mathrm{Tr}(ax) = a\mathrm{Tr}(x).$$

Aplicando la definición de función traza encontramos también que

$$(\mathrm{Tr}(x))^p = \mathrm{Tr}(x^p) = \mathrm{Tr}(x)$$

y, para $t = 2, \ldots, n$

$$(\mathrm{Tr}(x))^{p^t} = \mathrm{Tr}(x^{p^t}) = \mathrm{Tr}(x).$$

Esto nos permite afirmar que la traza de x tiene un orden que divide p y por tanto es un elemento de $\mathrm{GF}(p)$. En el campo $\mathrm{GF}(2^n)$, la traza de un elemento

esta en GF (2). Como la función traza es lineal sobre GF (2), se sigue que todas las funciones de la forma

$$f(a) = \mathrm{Tr}\,(wa)$$

son funciones 2-valuadas de GF (2^n), que son lineales sobre GF (2). Existen exactamente 2^n de tales funciones, una por cada valor de w. Llamaremos a la función Tr (wa) una *paridad de traza*, y a su correspondientes valor w *patrón de traza*. Dentro del análisis de las propiedades de correlación de las funciones sobre GF (2), las paridades de traza juegan un rol similar al que juegan las paridades en el análisis de correlación para funciones Booleanas. Como veremos mas adelante, cuando una representación es elegida, estas funciones pueden ser aplicadas uno-uno a las paridades.

Al trabajar con patrones de traza, es posible estudiar propiedades de correlación en funciones sobre GF (2^n) sin tener que especificar una base. De aquí que los resultados obtenidos son válidos para todas las elecciones de base. Una vez se ha elegido una base, los patrones de traza se pueden convertir en patrones de selección.

Para una función f sobre GF (2^n), denotamos la correlación entre una paridad de traza entrante Tr (wa) y una paridad de traza saliente Tr $(uf(a))$ por $C^f_{u,w}$. Así, tenemos

$$\begin{aligned} C^f_{u,w} &= 2^{-n} \sum_a (-1)^{\mathrm{Tr}\,(wa)} (-1)^{\mathrm{Tr}\,(uf(a))} \\ &= 2^{-n} \sum_a (-1)^{\mathrm{Tr}\,(wa)+\mathrm{Tr}\,(uf(a))} \\ &= 2^{-n} \sum_a (-1)^{\mathrm{Tr}\,(wa+uf(a))}. \end{aligned}$$

El valor de esta correlación está determinado por el número de valores a que satisfacen

$$\mathrm{Tr}\,(wa + uf(a)) = 0.$$

Si esta ecuación es satisfecha por r valores a, la correlación $C^f_{u,w}$ es igual a $2^{1-n}r$. Si esta no tiene soluciones, la correlación es -1; si esta es satisfecha para todos los valores de a, la correlación es 1; y si es satisfecha exactamente por la mitad de los posibles valores a, la correlación es 0. Empleando la expresión polinomial para f, ésta ecuación se convierte en la ecuación polinomial en a:

$$\mathrm{Tr}\left(wa + u\sum_i c_i a^i\right) = 0.$$

Como en el caso para propagación diferencial, para algunos casos el número de soluciones de estos polinomios puede ser determinado analíticamente proveyendo cotas verificables para las propiedades de correlación.

EJEMPLO. Consideremos la siguiente operación:

$$b = f(a) = a + c,$$

donde c es una constante. Una diferencia en a determina por completo la diferencia en b:

$$b' = f(a) + f(a + a') = (a + c) + (a + a' + c) = a'.$$

Por lo tanto la adición de una constante no tiene efecto sobre el patrón de diferencia. Para la correlación podemos encontrar el número de soluciones de

$$\mathrm{Tr}\,(wa + uf(a)) = 0,$$

que es equivalente a

$$\mathrm{Tr}\,(wa + ua + uc) = \mathrm{Tr}\,((\omega + u)a + uc) = 0.$$

Si $w + u$ es diferente de 0, la traza es cero para exactamente la mitad de los valores de a, y la correlación es 0. Si $w = u$ tenemos

$$\mathrm{Tr}\,(uc) = 0.$$

Esta ecuación es verdadera para todos los valores de a si $\mathrm{Tr}\,(uc) = 0$, y no tiene soluciones si $\mathrm{Tr}\,(uc) = 1$. Así, la adición de una constante no tiene efecto sobre el patrón de traza y además el signo de la correlación es igual a $(-1)^{\mathrm{Tr}\,(uc)}$.

6.1.3. Funciones que son Lineales sobre $\mathrm{GF}\,(2^n)$

Las funciones de $\mathrm{GF}\,(2^n)$ que son lineales sobre $\mathrm{GF}\,(2^n)$ son de la forma

$$f(a) = l^{(0)}a,$$

donde $l^{(0)}$ es un elemento de $\mathrm{GF}\,(2^n)$. Por lo que, hay exactamente 2^n funciones sobre $\mathrm{GF}\,(2^n)$ que son lineales sobre $\mathrm{GF}\,(2^n)$. Una diferencia en a determina por completo la diferencia en b:

$$b' = l^{(0)}a'.$$

Para la correlación podemos encontrar el número de soluciones de

$$\mathrm{Tr}\,(wa + uf(a)) = 0,$$

que esequivalente a

$$\mathrm{Tr}\,(wa + ul^{(0)}a) = \mathrm{Tr}\,((w + ul^{(0)})a) = 0,$$

Si el factor de a es diferente de 0, la correlación es cero. La correlación entre $\mathrm{Tr}\,(wa)$ y $\mathrm{Tr}\,(ub)$ es igual a 1 si

$$w = l^{(0)}u.$$

6.1.4. Funciones que son Lineales sobre $\mathrm{GF}(2)$

Una función f sobre $\mathrm{GF}(2^n)$ es lineal sobre $\mathrm{GF}(2)$ si satisface lo siguiente

$$\forall x, y \in \mathrm{GF}(2^n) : f(x+y) = f(x) + f(y)$$
$$\forall a \in \mathrm{GF}(2), \forall x \in \mathrm{GF}(2^n) : \mathrm{Tr}(ax) = a\mathrm{Tr}(x).$$

Observesé que una función que es lineal sobre $\mathrm{GF}(2^n)$ también lo es sobre $\mathrm{GF}(2)$; pero una función que es lineal sobre $\mathrm{GF}(2)$ no necesariamente es lineal sobre $\mathrm{GF}(2^n)$. Más aún, una función que satisface la primera condición de linealidad automáticamente satisfacerá la segunda condición. Como un ejemplo tenemos a la función $f(x) = x^2$ que es lineal sobre $\mathrm{GF}(2)$; pero no lo es sobre $\mathrm{GF}(2^n)$:

$$f(x+y) = (x+y)^2 = x^2 + xy + yx + y^2 = x^2 + y^2 = f(x) + f(y)$$
$$f(ax) = a^2 f(x) \neq af(x) \quad \text{si } a \notin \mathrm{GF}(2).$$

Esto puede ser extendido a todas las funciones de la forma $f(x) = x^{2^t}$. Es más, cualquier combinación lineal de estas funciones es lineal sobre $\mathrm{GF}(2)$. Se sigue que todas las funciones de la forma

$$f(a) = \sum_{t=0}^{n-1} l^{(t)} a^{2^t}, \text{ con } l^{(t)} \in \mathrm{GF}(2^n),$$

son lineales sobre $\mathrm{GF}(2)$. Además, nótese que todas las funciones de $\mathrm{GF}(2^n)$ que son lineales sobre $\mathrm{GF}(2)$ pueden ser representadas de esta manera (sección 3.4).

Usando la linealidad se concluye que una diferencia en a determina por completo la diferencia en b por:

$$b' = \sum_{t=0}^{n-1} l^{(t)} (a')^{2^t}.$$

La relación entre el patrón de traza entrante y el patrón de traza saliente es menos trivial.

TEOREMA 1. Para una función $b = \sum_{t=0}^{n-1} l^{(t)} a^{2^t}$ una traza de paridad saliente $\mathrm{Tr}(ub)$ está correlacionada con la traza de paridad entrante $\mathrm{Tr}(wa)$ en una correlación de 1 si, y sólo si

$$w = \sum_{t=0}^{n-1} (l^{(n-t \mod n)} u)^{2^t}. \tag{1}$$

Demostración. Probaremos que $\mathrm{Tr}(wa) = \mathrm{Tr}(ub)$ y en consecuencia $\mathrm{Tr}(wa + ub) = 0$ para todos los valores de a si w es de la forma dada. Todos los cálculos

que involucren las variables t, s y r se realizarán en módulo n, y todas las sumatorias van de 0 a $n-1$.

$$\begin{aligned}
\mathrm{Tr}\left(\sum_t (l^{(n-t)}u)^{2^t}a\right) &= \left(u\sum_t l^{(t)}a^{2^t}\right)\\
\sum_s\left(\sum_t l^{(n-t)^{2^t}}u^{2^t}a\right)^{2^s} &= \sum_s\left(\sum_t l^{(t)}ua^{2^t}\right)^{2^s}\\
\sum_s\sum_t l^{(n-t)^{2^{s+t}}}u^{2^{s+t}}a^{2^s} &= \sum_s\sum_t l^{(t)^{2^s}}u^{2^s}a^{2^{s+t}}\\
\sum_s\sum_t l^{(n-t)^{2^{s+t}}}u^{2^{s+t}}a^{2^s} &= \sum_{r=s+t}\sum_t l^{(t)^{2^{r-t}}}u^{2^{r-t}}a^{2^r}\\
\sum_s\sum_{r=n-t} l^{(r)^{2^{s-r}}}u^{2^{s-r}}a^{2^s} &= \sum_s\sum_t l^{(t)^{2^{s-t}}}u^{2^{s-t}}a^{2^s}\\
\sum_s\sum_t l^{(t)^{2^{s-t}}}u^{2^{s-t}}a^{2^s} &= \sum_s\sum_t l^{(t)^{2^{s-t}}}u^{2^{s-t}}a^{2^s}
\end{aligned}$$

□

6.2. Funciones sobre $(\mathrm{GF}(2^n))^l$

En esta sección trataremos las propiedades de la propagación diferencial y correlación de las funciones que operan sobre arreglos de l elementos de $\mathrm{GF}(2^n)$. Para representar estos arreglos usaremos la notación matricial

$$\mathbf{A} = [a_1\ a_2\ a_3\ \ldots\ a_l]^T.$$

donde los elementos $a_i \in \mathrm{GF}(2^n)$. Tenemos

$$F : (\mathrm{GF}(2^n))^l \to (\mathrm{GF}(2^n))^l, \quad \mathbf{A} \mapsto \mathbf{B} = F(\mathbf{A})$$

6.2.1. Propagación de la Diferencia

Una diferencia $\mathbf{A}'$ se propaga a una diferencia $\mathbf{B}'$ a través de F si, y sólo si

$$F(\mathbf{A}) + F(\mathbf{A} + \mathbf{A}') = \mathbf{B}'.$$

La probabilidad de la propagación de la diferencia $\mathrm{Prob}^F(\mathbf{A}', \mathbf{B}')$ es igual al número total de valores A que satisfacen esta ecuación, dividida por 2^{nl}.

6.2.2. Correlación

La paridad de traza puede extenderse vectorialmente. Puede definirse el patrón de traza de un vector como

$$\mathbf{W} = [w_1\ w_2\ w_3\ \ldots\ w_l]^T.$$

donde los elementos $w_i \in \mathrm{GF}(2^n)$. Las paridades de traza para un vector son entonces de la forma

$$\sum_i \mathrm{Tr}(w_i a_i) = \mathrm{Tr}\left(\sum_i w_i a_i\right) = \mathrm{Tr}(\mathbf{W}^T\mathbf{A}).$$

Podemos definir una correlación entre una paridad de traza entrante $\mathrm{Tr}(\mathbf{W}^T\mathbf{A})$ y una paridad de traza saliente $\mathrm{Tr}(\mathbf{U}^T\mathbf{A})$:

$$\begin{aligned} C^F_{\mathbf{U},\mathbf{W}} &= 2^{-nl}\sum_{\mathbf{A}}(-1)^{\mathrm{Tr}(\mathbf{W}^T\mathbf{A})}(-1)^{\mathrm{Tr}(\mathbf{W}^T F(\mathbf{A}))} \\ &= 2^{-nl}\sum_{\mathbf{A}}(-1)^{\mathrm{Tr}(\mathbf{W}^T\mathbf{A})+\mathrm{Tr}(\mathbf{W}^T F(\mathbf{A}))} \\ &= 2^{-nl}\sum_{\mathbf{A}}(-1)^{\mathrm{Tr}(\mathbf{W}^T\mathbf{A}+\mathbf{W}^T F(\mathbf{A}))}. \end{aligned}$$

6.2.3. Funciones que son Lineales sobre $\mathrm{GF}(2^n)$

Si F es lineal sobre $\mathrm{GF}(2^n)$, entonces puede denotarse por una multiplicación matricial, como

$$\mathbf{B} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_l \end{pmatrix} = \begin{pmatrix} l_{1,1} & l_{1,2} & l_{1,3} & \cdots & l_{1,l} \\ l_{2,1} & l_{2,2} & l_{2,3} & \cdots & l_{2,l} \\ l_{3,1} & l_{3,2} & l_{3,3} & \cdots & l_{3,l} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{l,1} & l_{l,2} & l_{l,3} & \cdots & l_{l,l} \end{pmatrix} \times \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_l \end{pmatrix} = \mathsf{L}\mathbf{A}$$

Los elementos de la matriz L son elementos de $\mathrm{GF}(2^n)$.
Una diferencia en $\mathbf{A}$ determina completamente la diferencia en $\mathbf{B}$:

$$\mathbf{B}' = \mathsf{L}\mathbf{A}'$$

Para la correlación tenemos:

$$\begin{aligned} \mathrm{Tr}(\mathbf{W}^T\mathbf{A} + \mathbf{U}^T\mathsf{L}\mathbf{A}) &= \mathrm{Tr}(\mathbf{W}^T\mathbf{A} + (\mathsf{L}^T\mathbf{U})^T\mathbf{A}) \\ &= \mathrm{Tr}((\mathbf{W} + \mathsf{L}^T\mathbf{U})^T\mathbf{A}). \end{aligned}$$

Por lo tanto, la correlación entre $\mathrm{Tr}(\mathbf{W}^T\mathbf{A})$ y $\mathrm{Tr}(\mathbf{U}^T\mathbf{A})$ es igual a 1 si

$$\mathbf{W} + \mathsf{L}^T\mathbf{U}$$

6.2.4. Funciones que son Lineales sobre $\mathrm{GF}(2)$

Generalizando la ecuación $b = f(a) = \sum_{t=0}^{n-1} l^{(t)} a^{2^t}$ (subsección 6.1.4) para vectores de $\mathrm{GF}(2^n)$, resulta

$$b_i = \sum_j \sum_t l_{i,j}^{(t)} a_j^{2^t}, \quad 0 \le i \le n.$$

Si introducimos la siguiente notación:

$$\mathbf{A}^{2^t} = \left(a_1^{2^t} \quad a_2^{2^t} \quad a_3^{2^t} \quad \cdots \quad a_l^{2^t}\right),$$

esto se puede escribir como

$$\mathbf{B} = \sum_t \mathsf{L}^{(t)} \mathbf{A}^{2^t}.$$

Debido a la linealidad, tenemos una vez más que una diferencia en A determina completamente la diferencia en B por

$$\mathbf{B}' = \sum_t \mathsf{L}^{(t)} (\mathbf{A}')^{2^t}.$$

Para la relación entre el patrón de traza entrante y el patrón de traza saliente, establecemos lo siguiente:

$$\mathbf{W} = \sum_t \left((\mathsf{L}^{(n-t \mod n)})^T \mathbf{U} \right)^{2^t}.$$

Capítulo 7

Representaciones de $\mathrm{GF}(p^n)$

El presente capítulo estará enfocado al estudio de las diferentes representaciones de $\mathrm{GF}(p^n)$. Primero será descrita la representación cíclica que simplifica la operación de multiplicación. A esta le seguirá el tratamiento de la representación vectorial de $\mathrm{GF}(p^n)$ y las bases duales, que cumplen un papel esencial en el propósito de establecer una correspondencia entre las propiedades de propagación inherentes a las funciones sobre $\mathrm{GF}(2^n)$ y aquéllas pertenecientes a las funciones Booleanas.

7.1. Representación cíclica de $\mathrm{GF}(p^n)$

Afirmamos que el grupo multiplicativo $\mathrm{GF}(p^n)^*$ es cíclico (véase el Teorema 3 en la sección 3.1). Los elementos de este grupo (diferentes de 0) pueden ser representados como la (p^n-1)-ésima potencia de un generador $\alpha \in \mathrm{GF}(p^n)$:

$$\forall x \in \mathrm{GF}(p^n)\backslash\{0\}, \exists a_x \in \mathbb{Z}_{p^n-1}: \quad x = \alpha^{a_x}.$$

En esta representación, la multiplicación de dos elementos no nulos corresponde a la adición de sus potencias módulo p^n-1:

$$x \cdot y = \alpha^{a_x} \cdot \alpha^{a_y} = \alpha^{a_x + a_y \mod p^n-1}. \tag{1}$$

Operaciones como calcular el inverso multiplicativo y la potenciación se hacen triviales en esta representación. Para la adición sin embargo, la representación vectorial es la más apropiada. En cálculos que involucren adición y multiplicación, se puede trocar entre las dos representaciones por medio de las tablas de conversión. La tabla usada para la conversión de la representación vectorial a la representación cíclica es llamada *tabla log* y la tabla para la conversión inversa es llamada *tabla alog*.

7.2. Representación Vectorial de $\mathrm{GF}(p^n)$

Los elementos de un campo finito $\mathrm{GF}(p^n)$ pueden ser representados como vectores pertenecientes a un espacio vectorial n-dimensional sobre $\mathrm{GF}(p)$, comúnmen-

te denotado por GF$(p)^n$. La adición de vectores en este espacio vectorial corresponde a la adición en GF(p^n). Podemos elegir una base consistente de n elementos $e^{(i)} \in$ GF(p^n). Representaremos una base e por un vector columna que tenga como posiciones los elementos de la base:

$$\mathsf{e} = \begin{pmatrix} e^{(1)} & e^{(2)} & \cdots & e^{(n)} \end{pmatrix}^T.$$

Los elementos de GF(p^n) pueden ser representados por sus coordenadas con respecto de esta base. Tenemos

$$a = \sum_i a_i e^{(i)} = \mathbf{a}^T \mathsf{e}, \tag{2}$$

donde los a_i son las coordenadas de a con respecto a la base e, y a es el vector columna consistente de las coordenadas de a_i. Las coordenadas son elementos de GF(p). Dada una base, existe una correspondencia uno-uno entre los elementos del campo GF(p^n) y sus coordenadas. El elemento cero del campo GF(p^n), denotado por $\mathbf{0}$, tiene todas sus coordenadas iguales al elemento cero del campo GF(p):

$$\mathbf{0} = \begin{pmatrix} 0 & 0 & \cdots & 0 \end{pmatrix}.$$

Las coordenadas de la suma de dos vectores están dadas por el vector suma de las coordenadas de los dos vectores:

$$\mathbf{d} = \mathbf{b} + \mathbf{c} \quad \Longleftrightarrow \quad d_i = b_i + c_i, 0 < i \leq n.$$

Aquí, la suma de los coeficientes ocurre en el campo GF(p). Se sigue que las coordenadas del elemento inverso (aditivo) de b pueden ser calculados reemplazando cada coordenada por su inverso (aditivo) en GF(p). En un campo finito de característica 2, cada coordenada es su propio elemento inverso respecto a la adición, lo cual tiene lugar para cada elemento del campo. La representación polinomial es un caso especial de la representación vectorial. La base en este caso es de la forma $1, x, x^2, x^3, \ldots, x^{n-1}$.

7.3. Bases Duales

Las coordenadas de un elemento del campo con respecto a una base, pueden expresarse en términos de la base dual y de la función traza.

DEFINICIÓN. Dos bases e y d son llamadas bases duales, si para todo i y j con $1 \leq i, j \leq n$, se cumple que

$$\text{Tr}\,(d^{(i)} e^{(j)}) = \delta(i \oplus j). \tag{3}$$

Donde $\oplus$ denota la suma en GF(2). Cada base tiene exactamente una base dual. Si e y d son bases duales, entonces tenemos

$$\text{Tr}\,(d^{(j)} a) = \text{Tr}\left(d^{(j)} \sum_{i=1}^{n} a_i e^{(i)} \right) = \sum_{i=1}^{n} a_i \text{Tr}\,(d^{(j)} e^{(i)}) = a_j.$$

De aquí que las coordenadas con respecto a la base e pueden ser expresadas de forma elegante por medio de la función traza y la base dual d:

$$\mathbf{a} = \begin{pmatrix} \mathrm{Tr}\,(d^{(1)}a) & \mathrm{Tr}\,(d^{(2)}a) & \cdots & \mathrm{Tr}\,(d^{(n)}a) \end{pmatrix}. \tag{4}$$

Aplicando (2) tenemos

$$a = \sum_{i=1}^{n} \mathrm{Tr}\,(d^{(i)}a)e^{(i)} = \sum_{i=1}^{n} \mathrm{Tr}\,(e^{(i)}a)d^{(i)}.$$

Capítulo 8

Funciones Booleanas y Funciones en $\mathrm{GF}(2^n)$

Las funciones de $\mathrm{GF}(2^n)$ pueden ser aplicadas a funciones de $\mathrm{GF}(2)^n$ eligiendo una base e en $\mathrm{GF}(2^n)$. Dada

$$f : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n), \quad a \mapsto b = f(a),$$

podemos definir una función Booleana ϕ:

$$\phi : \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n, \quad \mathbf{a} \mapsto \mathbf{b} = \phi(\mathbf{a}),$$

donde $\mathbf{a} = \begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} b_1 & b_2 & \cdots & b_n \end{pmatrix}$ y $a_i = \mathrm{Tr}(ad^{(i)})$, $b_i = \mathrm{Tr}(bd^{(i)})$.
Por otro lado, dada una función Booleana ϕ, una función sobre $\mathrm{GF}(2^n)$ puede ser definida como

$$a = \mathbf{a}^T \mathsf{e}$$
$$b = \mathbf{b}^T \mathsf{e}.$$

Pudiendo ésto extenderse a funciones operando sobre vectores de elementos de $\mathrm{GF}(2^n)$.

8.1. Diferencias en $\mathrm{GF}(2)^n$ y $\mathrm{GF}(2^n)$

Ahora podemos considerar como un patrón de diferencia en $\mathrm{GF}(2)^n$ se puede hacer corresponder a un patrón de diferencia en $\mathrm{GF}(2^n)$. Gracias a la linealidad de la función traza, la correspondencia entre un patrón de diferencia a' en $\mathrm{GF}(2^n)$ y un patrón de diferencia $\mathbf{a}'$ en $\mathrm{GF}(2)^n$ esta dada por

$$\mathbf{a}' = \begin{pmatrix} a'_1 & a'_2 & \cdots & a'_n \end{pmatrix},$$

donde

$$a'_i = \mathrm{Tr}(a' d^{(i)}) \quad \text{y} \quad a' = \mathbf{a}'^T \mathsf{e}$$

8.2. Relación entre Patrones de Traza y Patrones de Selección

En el siguiente teorema probaremos que para cada patrón de selección existe un patrón de traza que le corresponde. Por lo tanto, cuando estudiemos la propagación de correlaciones, podemos usar patrones de traza. Es así que evitaremos la especificación de una base, que sería necesaria si se emplean patrones de selección.

Sean las coordenadas de a con respecto a e por $\mathbf{a}$, y las coordenadas de w con respecto a d por $\mathbf{w}_\mathsf{d}$, donde d y e son bases duales.

TEOREMA 1. La relación entre un patrón de traza y el correspondiente patrón de selección está dada por

$$\mathrm{Tr}\,(wa) = \mathbf{w}_\mathsf{d}^T \mathbf{a}.$$

Demostración. Aplicando la expresión de las coordenadas de un elemento del campo con respecto a dos bases duales a w y a, conseguimos

$$\mathrm{Tr}\,(wa) = \mathrm{Tr}\left(\left(\sum_i \mathrm{Tr}\,(e^{(i)}w)d^{(i)}\right)\left(\sum_j \mathrm{Tr}\,(d^{(j)}a)e^{(j)}\right)\right).$$

Como la imagen de un elemento por la aplicación traza está en $\mathrm{GF}\,(2)$, y como la aplicación traza es lineal sobre $\mathrm{GF}\,(2)$, podemos convertir esto a:

$$\begin{aligned}\mathrm{Tr}\,(wa) &= \sum_i \mathrm{Tr}\,(e^{(i)}w)\sum_j \mathrm{Tr}\,(d^{(j)}a)\mathrm{Tr}\,(d^{(i)}e^{(j)}) \\ &= \sum_i \mathrm{Tr}\,(e^{(i)}w)\sum_j \mathrm{Tr}\,(d^{(j)}a)\delta(i\oplus j) \\ &= \sum_i \mathrm{Tr}\,(e^{(i)}w)\mathrm{Tr}\,(d^{(i)}a).\end{aligned}$$

Y aplicando (4) dos veces se completa la demostración. □

8.3. Relación entre Funciones Lineales en $\mathrm{GF}\,(p)^n$ y en $\mathrm{GF}\,(p^n)$

Una función lineal de $\mathrm{GF}\,(p)^n$ queda completamente especificada por una matriz M de tamaño $n \times n$:

$$\mathbf{b} = \mathsf{M}\mathbf{a}$$

Una función lineal de $\mathrm{GF}\,(p^n)$ está especificada por los n coeficientes $l^{(t)} \in \mathrm{GF}\,(p)^n$ en

$$b = \sum_{t=0}^{n-1} l^{(t)} a^{p^t}.$$

Después de elegir una base e sobre GF (p^n), estas dos representaciones pueden convertirse una en otra.

TEOREMA 2. Dados los coeficientes $l^{(t)}$ y una base e, los elementos de la matriz M están dados por

$$M_{ij} = \sum_{t=0}^{n-1} \text{Tr}\left(l^{(t)} d^{(i)} e^{(j)^{p^t}}\right).$$

Demostración. Derivaremos una expresión de b_i como una combinación lineal de a_j en términos de los factores $l^{(t)}$. Para una componente b_i tenemos

$$b_i = \text{Tr}\left(b d^{(i)}\right) = \text{Tr}\left(\sum_t l^{(t)} a^{p^t} d^{(i)}\right) = \sum_t \text{Tr}\left(l^{(t)} a^{p^t} d^{(i)}\right). \tag{5}$$

Las potencias de a pueden ser expresadas en términos de las componentes a_j:

$$a^{p^t} = \left(\sum_j a_j e^{(j)}\right)^{p^t} = \sum_j a_j e^{(j)^{p^t}}, \tag{6}$$

donde usamos el hecho de que la exponenciación por p^t es lineal sobre GF (p) para obtener (6). Entonces sustituyendo (6) en (5) se obtiene

$$\begin{aligned} b_i &= \sum_t \text{Tr}\left(l^{(t)} \sum_j a_j e^{(j)^{p^t}} d^{(i)}\right) \\ &= \sum_t \sum_j \text{Tr}\left(l^{(t)} e^{(j)^{p^t}} d^{(i)} a_j\right) \\ &= \sum_j \left(\sum_t \text{Tr}\left(l^{(t)} e^{(j)^{p^t}} d^{(i)}\right)\right) a_j. \end{aligned}$$

De donde se sigue que

$$M_{ij} = \sum_t \text{Tr}\left(l^{(t)} e^{(j)^{p^t}} d^{(i)}\right),$$

probando el teorema. □

TEOREMA 3. Dados la matriz M y una base e, los elementos $l^{(t)}$ están dados por

$$l^{(t)} = \sum_{i=1}^{n} \sum_{j=1}^{n} M_{ij} d^{(j)^{p^t}} e^{(i)}.$$

Demostración. Expresaremos b como una función de potencias de a en términos de los elementos de la matriz M. Tenemos

$$b = \sum_i b_i e^{(i)}, \tag{7}$$

y

$$\begin{aligned} b_i &= \sum_j M_{ij} a_j \\ &= \sum_j M_{ij} \mathrm{Tr}\left(ad^{(j)}\right) \\ &= \sum_j M_{ij} \sum_t a^{p^t} d^{(j)^{p^t}}. \end{aligned} \tag{8}$$

Sustituyendo (8) en (7) se obtiene

$$\begin{aligned} b &= \sum_i \sum_j M_{ij} \sum_t a^{p^t} d^{(j)^{p^t}} e^{(i)} \\ &= \sum_t \left(\sum_i \sum_j M_{ij} d^{(j)^{p^t}} e^{(i)} \right) a^{p^t}. \end{aligned}$$

De donde se sigue que

$$l^{(t)} = \sum_i \sum_j M_{ij} d^{(j)^{p^t}} e^{(i)},$$

probando el teorema. □

8.4. Ilustración

Las siguiente Figura conjuntamente con el ejemplo a continuación, ilustran los resultados para la propagación de los rastros lineales a través de las funciones lineales de GF(p^n). Recuerdesé que siempre expresaremos el patrón *entrante* w como una función del patrón *saliente* u.

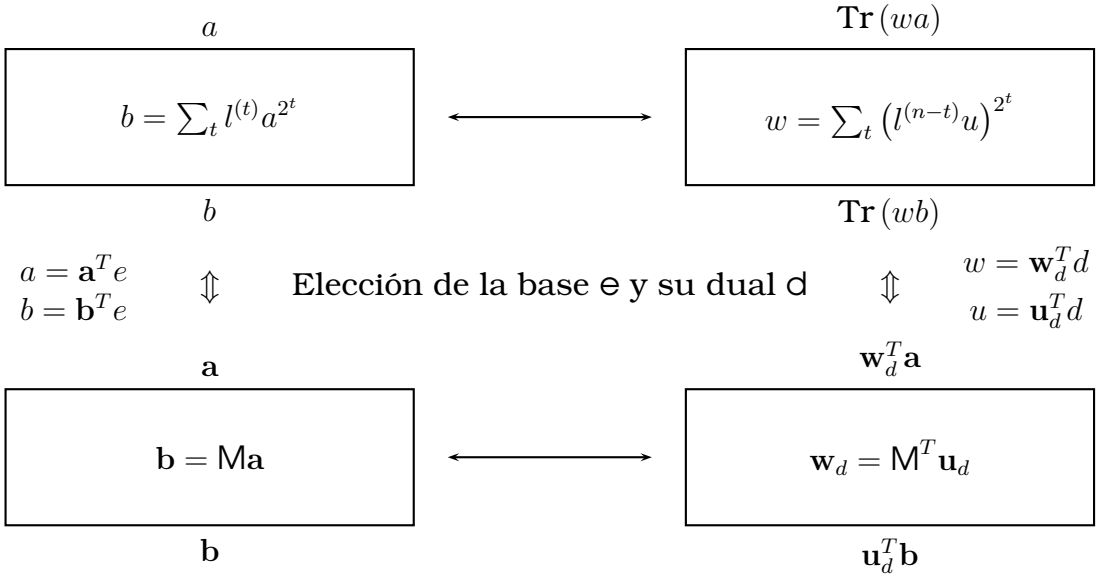

Figura 1. Propagación del rastro lineal por medio de una función lineal.

EJEMPLO. Consideremos el campo GF (2^3), con α como una raíz de $x^3+x+1=0$. Entonces los elementos de GF (2^3) pueden ser denotados por 0, 1, α, $\alpha+1$, α^2, α^2+1, $\alpha^2+\alpha$ y $\alpha^2+\alpha+1$. Consideremos dos transformaciones f y g, definidas por

$$f(a) = \alpha a$$
$$g(a) = a^4 + (\alpha^2+\alpha+1)a^2$$

Para ambas funciones, queremos derivar una expresión que para cualquier patrón de traza saliente u dado nos de el patrón de traza entrante w correlacionado con aquél. Denotaremos esta expresión por f_d y g_d, respectivamente, donde el subíndice d hace referencia a *dual*. Podemos derivar f_d y g_d sin tener que considerar temas de representaciones, debido a la aplicación del Teorema 1 de la subsección 6.1.4.

Para $f(a)$ tenemos

$$l^{(0)} = \alpha, \quad l^{(1)} = l^{(2)} = 0,$$

y por tanto

$$w = f_d(u) = \alpha u. \tag{9}$$

Para $g(a)$ tenemos

$$l^{(0)} = 0, \quad l^{(1)} = \alpha^2+\alpha+1, \quad l^{(2)} = 1,$$

y de aquí se tiene que

$$w = g_d(u) = u^2 + ((\alpha^2+\alpha+1)u)^4 = u^2 + (\alpha^2+1)u^2. \tag{10}$$

Ahora bien, alternativamente podemos elegir una base y trabajar con la representación vectorial. Comenzamos entonces eligiendo la base:

$$\mathsf{e} = \begin{pmatrix} \alpha^2+\alpha+1 & \alpha+1 & 1 \end{pmatrix}^T.$$

La siguiente tabla muestra las coordenadas de los elementos de GF (2^3), así como las imágenes de f y g con respecto a esta base e

x	$\mathbf{x}$	$\mathbf{y} = f(x)$	$\mathbf{y} = g(x)$
0	000	000	000
1	001	011	101
$\alpha+1$	010	101	001
α	011	110	100
$\alpha^2+\alpha+1$	100	111	100
$\alpha^2+\alpha$	101	100	001
α^2	110	010	101
α^2+1	111	001	000

Coordenadas de los elementos del campo y de las imágenes de f y g con respecto a la base e.

Una vez que las coordenadas de las entradas y salidas de f y g han sido determinadas, podemos derivar las matrices M y N que describan las funciones f y g en el espacio vectorial:

$$\mathsf{M} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \mathsf{N} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}.$$

Entonces, las transformaciones para derivar patrones de selección entrantes a partir de los patrones de selección salientes estarán determinadas por M^T y N^T

$$\begin{aligned} f_d(\mathbf{u}_d) &= \mathsf{M}^T \mathbf{u}_d \\ g_d(\mathbf{u}_d) &= \mathsf{N}^T \mathbf{u}_d. \end{aligned} \tag{11}$$

La siguiente tabla muestra las coordenadas de los elementos de GF(2^3), así como las coordenadas de las imágenes de f_d y g_d calculadas de acuerdo con las ecuaciones en (11).

$\mathbf{u}_d$	$\mathbf{w}_d = f(\mathbf{u}_d)$	$\mathbf{w}_d = g(\mathbf{u}_d)$
000	000	000
001	111	011
010	101	000
011	010	011
100	110	101
101	001	110
110	011	101
111	100	110

Las funciones f_d y g_d.

La base dual de e puede ser determinada resolviendo (3). Esta estará dada por

$$\mathsf{d} = \begin{pmatrix} \alpha & \alpha^2 + \alpha & \alpha^2 + 1 \end{pmatrix}^T.$$

Puede ahora verificarse que las coordenadas dadas en la última tabla, corresponden a las definiciones de f_d y g_d de las ecuaciones (9) y (10), probando así que las coordenadas de las variables u y w están determinadas respecto a la base d.

Capítulo 9

Rijndael-GF

Las transformaciones de ronda del algoritmo Rijndael operan sobre un estado en $\mathrm{GF}(2)^{8n_t}$ donde $n_t \in \{16, 20, 24, 28, 32\}$. Ahora describiremos cómo cada uno de sus pasos puede ser generalizado a operaciones en $\mathrm{GF}(2^8)^{n_t}$. La etapa no-lineal **`SubBytes`** opera sobre los elementos individuales de el estado. Esta está compuesta de dos pasos. El primer paso es tomar la inversa multiplicativa en $\mathrm{GF}(2^n)$:

$$g(x) = x^{-1}, \tag{12}$$

donde los bytes representan polinomios.

El siguiente paso es la aplicación de una función afín definida por una multiplicación matricial y la subsecuente adición de una constante. Si Aplicamos el Teorema 3 del capítulo 8, obtendremos la siguiente expresión para la función afín:

$$\begin{aligned} f(x) = {} & \mathbf{05} \cdot x + \mathbf{09} \cdot x^2 + \mathbf{F9} \cdot x^{2^2} + \mathbf{25} \cdot x^{2^3} + \mathbf{F4} \cdot x^{2^4} + \\ & + \mathbf{01} \cdot x^{2^5} + \mathbf{B5} \cdot x^{2^6} + \mathbf{8F} \cdot x^{2^7} + \mathbf{63}. \end{aligned} \tag{13}$$

Componiendo (12) y (13) resulta la expresión de la S-box del Rijndael-GF:

$$\begin{aligned} f_{RD}(x) = {} & \mathbf{05} \cdot x^{254} + \mathbf{09} \cdot x^{253} + \mathbf{F9} \cdot x^{251} + \mathbf{25} \cdot x^{247} + \mathbf{F4} \cdot x^{239} + \\ & + \mathbf{01} \cdot x^{223} + \mathbf{B5} \cdot x^{191} + \mathbf{8F} \cdot x^{127} + \mathbf{63}. \end{aligned} \tag{14}$$

En esta expresión los coeficientes son elementos de $\mathrm{GF}(2^8)$. Denotaremos este polinomio por $f_{RD}(x)$. Para la generalización de la etapa **`SubBytes`** tenemos que reemplazar cada elemento del estado s_i por $f_{RD}(s_i)$.

La etapa **`ShiftRows`** es una transposición de bytes que no modifica los valores; sino que meramente cambia su orden. Por lo tanto, su generalización también cambiara el orden de los elementos de s_i del estado sin cambiar su valor. La etapa de mezcla, **`MixColumns`**, opera independientemente sobre las columnas de 4 bytes y sólo las mezcla linealmente. Ha sido ya definida esta etapa como la multiplicación por una matriz sobre vectores de 4 elementos de $\mathrm{GF}(2^8)$ que definen completamente tal generalización (véase el capítulo 4). La adición de la clave de ronda **`AddRoundKey`** consiste de una simple operación XOR sobre los bits, que se corresponde con la adición en todas las representaciones vectoriales de $\mathrm{GF}(2^8)$.

La expansión de clave sólo hace uso de la operación XOR bit con bit, transposicion de bytes, la S-box de Rijndael y una operación adicional XOR a nivel de bits, involucrando constantes de ronda. Con respecto a las primeras 3 operaciones señaladas, ya se expuso cómo éstas han de ser generalizadas. Las constantes de ronda han sido a su vez definidas en términos de elementos de $\mathrm{GF}(2^8)$ (véase nuevamente el capítulo 4).

Rijndael-GF, conjuntamente con la elección de una representación de elementos de $\mathrm{GF}(2^8)$ como bytes, constituyen un cifrado por bloques. Por ejemplo, el cifrado Rijndael es la representación de Rijndael-GF donde los elementos de $\mathrm{GF}(2^8)$ son codificados como bytes denotándolos por medio de polinomios binarios de grado menor que 8, donde la multiplicación del campo está definida módulo el polinomio irreducible $m(x) = x^8 + x^4 + x^3 + x + 1$.

Capítulo 10

Descripción Polinomial del Rijndael AES

El principal interés acá, es el de describir el algoritmo Rijndael de una forma algebraicamente transparente. En este capítulo final se mostrará cómo el algoritmo en su totalidad, puede ser descrito de un modo bastante elegante a través de una secuencia de manipulaciones algebraicas en un anillo finito. Todo el tratamiento derivará en una descripción de la estructura de la S-box y el cálculo de los coeficientes de su respectiva función que representa la parte no lineal principal del cifrado Rijndael. Tales coeficientes dados en notación polinomial, coinciden con aquéllos derivados en el capítulo anterior presentados allá en notacion hexadecimal.

10.1. Rijndael bajo el Enfoque de la Teoría de Anillos

Sea $\mathbb{Z}_2 = \{0, 1\}$ el campo binario y consideremos el polinomio irreducible

$$\mu(z) = z^8 + z^4 + z^3 + z + 1 \in \mathbb{Z}_2[z].$$

Sea $\mathbb{F} = \mathbb{Z}_2/\langle \mu(z) \rangle = \mathrm{GF}\,(256)$ el campo de Galois de 2^8 elementos y consideremos el ideal:

$$I = \langle x^4 + 1, y^4 + 1, \mu(z) \rangle \subset \mathbb{Z}_2[x, y, z].$$

Describiremos el algoritmo Rijndael por medio de una sucesión de manipulaciones de polinomios en el anillo finito

$$R = \mathbb{Z}_2[x, y, z]/I = \mathbb{F}[x, y]/\langle x^4 + 1, y^4 + 1 \rangle.$$

El anillo R tiene simultáneamente la estructura de un $\mathbb{Z}_2$-álgebra finita y la estructura de un $\mathbb{F}$-álgebra finita como claramente se ve en esta descripción. Los monomios

$$\{x^i y^j z^k |\ 0 \leq i, j \leq 3,\ 0 \leq k \leq 7\}$$

forman una $\mathbb{Z}_2$-base del anillo R como un álgebra. En particular $\dim_{\mathbb{Z}_2} R = 128$, es decir $|R| = 2^{128}$. Los cálculos en el anillo R pueden hacerse muy eficientemente. La adición en R se lleva a cabo componente a componente y la multiplicación en R se realiza a través de la multiplicación en $\mathbb{Z}_2[x, y, z]$ seguida por la reducción módulo el ideal I.

Aclaración. Fácilmente se verifica que $x^4 + 1, y^4 + 1, \mu(z)$ forman una base reducida de Gröbner (consúltese [17] y [2]) del ideal I que también es un ideal 0-dimensional. Como consecuencia, la reducción módulo I es muy fácil de lograr.

Siempre que $r \in R$ sea un elemento, definiremos los elementos $r_{i,j} \in \mathbb{F}$ y $r_j \in \mathbb{F}[x]/\langle x^4 + 1\rangle$ por medio de:

$$r = \sum_{i=0}^{3}\sum_{j=0}^{3} r_{i,j}x^i y^j = \sum_{i=0}^{3}\left(\sum_{j=0}^{3} r_{i,j}x^i\right) y^j = \sum_{i=0}^{3} r_j y^j.$$

Ahora bien, en un nivel abstracto, un criptosistema de clave secreta consiste de un espacio de mensajes o texto-franco M; un espacio de texto-cifrado C; y un espacio de claves K junto a una aplicación encriptadora

$$\epsilon : M \times K \to C$$

y una aplicación desencriptadora

$$\delta : C \times K \to M$$

tal que $\delta(\epsilon(m, k), k) = m$ para todo $m \in M$ y $k \in K$. Tómese en cuenta algo crucial, que no debe ser factible el cómputo de la clave secreta $k \in K$ a partir de una sucesión de parejas de (texto-franco,texto-cifrado) $(m^{(t)}, c^{(t)} = \epsilon(m^{(t)}, k))$, $t = 1, 2, \ldots$

Recordemos que en el sistema de cifrado Rinjdael AES, se tiene la posibilidad de trabajar con claves secretas de 128 bits, 192 bits o 256 bits. Describiremos el sistema cuando $|K| = 2^{128}$ e indicaremos más adelante cómo adaptar la descripción algebraica a otras situaciones. Para el algoritmo Rijndael definiremos

$$K = M = C = R.$$

Crucial para la descripción será el siguiente polinomio que aparece en la propuesta original de Daemen y Rijmen:

$$\begin{aligned}\phi(u) = {} & (z^2+1)u^{254} + (z^3+1)u^{253} + (z^7+z^6+z^5+z^4+z^3+1)u^{251} + \\ & + (z^5+z^2+1)u^{247} + (z^7+z^6+z^5+z^4+z^2)u^{239} + u^{223} + (z^7+z^5+z^4+z^2+1)u^{191} + \\ & + (z^7+z^3+z^2+z+1)u^{127} + (z^6+z^5+z+1) \in \mathbb{F}[u].\end{aligned}$$

Asumamos que Alicia y Roberto comparten una clave secreta $k \in R$ y Alicia quiere encriptar el mensaje $m \in R$. En una primera etapa Alicia y Roberto expanden la clave resultando así 11 elementos $k^{(t)} \in R$, $t = 0, 1, \ldots, 10$.

Expansión de la Clave: Alicia y Roberto calculan recursivamente 11 elementos $k^{(t)} \in R$, $t = 0, 1, \ldots, 10$ de la siguiente manera:

$$\begin{aligned} k^{(0)} &= k \\ k_0^{(t+1)} &= \left(\sum_{i=0}^{3} \phi(k_{i,3}^{(t)}) x^i \right) x^3 + z^t + k_0^{(t)}, \quad \text{para } t = 0, 1, \ldots, 9 \\ k_i^{(t+1)} &= k_{i-1}^{(t+1)} + k_i^{(t)} \quad \text{para } t = 0, 1, \ldots, 9,\ i = 1, 2, 3. \end{aligned}$$

Con el propósito de describir el algoritmo de encriptación, definimos el elemento del anillo

$$\gamma = (z+1)x^3 + x^2 + x + z \in R$$

Algoritmo de Encriptación Rijndael: Usando las claves de ronda $k^{(t)} \in R$ y comenzando con el mensaje $m \in R$ Alicia calcula recursivamente:

$$\begin{aligned} m^{(0)} &= m + k^{(0)} \\ m^{(t+1)} &= \gamma \sum_{i=0}^{3} \sum_{j=0}^{3} \phi(m_{i,j}^{(t)}) x^i y^{3i+j} + k^{(t+1)}, \quad \text{para } t = 0, 1, \ldots, 8 \\ c = m^{(10)} &= \sum_{i=0}^{3} \sum_{j=0}^{3} \phi(m_{i,j}^{(9)}) x^i y^{3i+j} + k^{(10)}. \end{aligned}$$

El texto-cifrado transmitido por Alicia es c. Nótese que en la décima ronda no se multiplica por γ. Esto permite asegurar que la encriptación y desencriptación pueden formalmente ser descritas por las mismas operaciones algebraicas. A continuación describiremos el proceso de decencriptación.

Algoritmo de Desencriptación Rijndael: El polinomio ϕ es un polinomio de permutación que permuta los elementos de $\mathbb{F}$.

Existe un único polinomio de permutación $\psi(u) \in \mathbb{F}[u]$ de grado a lo sumo 255 tal que $\phi \circ \psi = \psi \circ \phi = Id_{\mathbb{F}}$ y derivaremos este polinomio más adelante. El elemento $\gamma \in R$ es invertible con

$$\gamma^{-1} = (z^3 + z + 1)x^3 + (z^3 + z^2 + 1)x^2 + (z^3 + 1)x + (z^3 + z^2 + z) \in R.$$

Usando la aplicación ψ, el elemento γ^{-1} y las claves de ronda $k^{(t)}$ Roberto puede descifrar el mensaje m de Alicia por medio del esquema:

$$\begin{aligned} c^{(0)} &= c + k^{(10)} \\ c^{(t+1)} &= \gamma^{-1} \sum_{i=0}^{3} \sum_{j=0}^{3} \psi(c_{i,j}^{(t)}) x^i y^{i+j} + \gamma^{-1} k^{(9-t)}, \quad \text{para } t = 0, 1, \ldots, 8. \\ c^{(10)} &= \sum_{i=0}^{3} \sum_{j=0}^{3} \psi(c_{i,j}^{(9)}) x^i y^{i+j} + k^{(0)}. \end{aligned}$$

Se verifica facilmente que $m = c(10)$. Nótese que el esquema de encriptación y el esquema de desencriptación siguen formalmente la misma secuencia de transformaciones. Basta cambiar ϕ por ψ; el producto con γ por la multiplicación por

γ^{-1} y el esquema de programación de las claves es modificado reemplazando $k^{(t)}$, $t = 0, \ldots, 10$ con $k^{(10)}, \gamma^{-1}k^{(9)}, \ldots, \gamma^{-1}k^{(1)}, k^{(0)}$.

Aclaración. Ambos procesos, tanto el de encriptación como el de desencriptación pueden realizarse de manera muy eficiente. En la practica, los polinomios ϕ y ψ no se evalúan, pues en su lugar se emplean tablas que consignan los valores arrojados por la aplicación de las permutaciones $\phi, \psi : \mathbb{F} \to \mathbb{F}$. La sustitución de exponentes $x^i y^j \mapsto x^i y^{3i+j}$ no requiere demasiado cálculos aritmético, y la adición de una clave de ronda $k^{(t+1)}$ se realiza eficazmente empleando la operación Booleana XOR. Los cálculos aritméticos pueden requerirse cuando se multiplica por γ y respectivamente por γ^{-1}. Esto también puede mostrarse por medio de tablas. Si se utilizan operaciones algebraicas, entonces la multiplicación por γ es ligeramente mas facil que la multiplicación por γ^{-1} y por ello el algoritmo de desencriptación es ligeramente mas complejo que el algoritmo de encriptación.

Aclaración. γ fue elegido de tal manera que la multiplicación por γ se pueda realizar con un mínimo número de ramal y al mismo tiempo se garantice una buena difusión de $\mathbb{F}[x]/\langle x^4 + 1\rangle$. Sin embargo, debido al pequeño orden de γ en R, ésto último no goza del convencimiento de que la elección de γ fuera óptima. Un cálculo directo muestra que γ tiene orden cuatro. Así tendremos una sencilla expresión para γ^{-1}:

$$\gamma^{-1} = \gamma^3 = \gamma^2\gamma = (z^2x^2 + z^2 + 1)\gamma.$$

En lugar de multiplicar por γ^{-1} es por lo tanto posible, multiplicar tres veces por γ o alternativamente, se puede multiplicar por γ y después multiplicar por $(z^2x^2 + z^2 + 1)$. Esto es más eficiente que multiplicar por expresión completa de γ^{-1}.

10.2. Relación con la Descripción Estándar

En la descripción original del algoritmo Rijndael el anillo R no es utilizado. En su lugar, conjuntos de elementos conteniendo 128 bits fueron configurados en un arreglo de 4×4 donde cada posición contiene un byte, es decir 8 bits. Pensando en relacionar ambas descripciones, asignaremos a cada elemento $r = \sum_{i=0}^{3}\sum_{j=0}^{3} r_{i,j}x^i y^j$ el arreglo 4×4

$r_{0,0}$	$r_{0,1}$	$r_{0,2}$	$r_{0,3}$
$r_{1,0}$	$r_{1,1}$	$r_{1,2}$	$r_{1,3}$
$r_{2,0}$	$r_{2,1}$	$r_{2,2}$	$r_{2,3}$
$r_{3,0}$	$r_{3,1}$	$r_{3,2}$	$r_{3,3}$

donde cada elemento $r_{i,j} \in \mathbb{F}$ se visualiza como un byte. Usando un esquema específico las siguientes operaciones serán perfiladas:

Transformación `SubBytes` S-box: En esta operación cada elemento $r_{i,j} \in \mathbb{F}$ es modificado usando una permutación ϕ del grupo simétrico de 256 elementos.

La permutación ϕ se descompone en las siguientes tres permutaciones:

$$\begin{aligned}
\phi_1 : \mathbb{F} \to \mathbb{F}, \quad & f \mapsto \begin{cases} f^{-1} & , \text{ si } f \neq 0 \\ 0 & , \text{ si } f = 0 \end{cases}, \\
L : \mathbb{F} \to \mathbb{F}, \quad & f \mapsto (z^4 + z^3 + z^2 + z + 1)f \mod z^8 + 1, \\
\phi_3 : \mathbb{F} \to \mathbb{F}, \quad & f \mapsto z^6 + z^5 + z + 1 + f.
\end{aligned}$$

La permutación ϕ está definida como $\phi = \phi_3 \circ L \circ \phi_1$. Es posible describir la permutación ϕ por medio de polinomios de permutación. Para ésto, nótese que cada permutación de $\mathbb{F}$ puede también ser descrita por medio de un único polinimio de interpolación (un elemento de $\mathbb{F}[u]$) de grado a lo sumo 255. Denotaremos este único polinomio por $\phi(u)$. El contexto siempre dejará en claro si vemos a ϕ como una permutación o como un polinomio $\phi(u) \in \mathbb{F}[u]$.

El único polinomio de permutación $\phi(u)$ puede ser computado de la siguiente manera. Si $\alpha \neq 0$ entonces

$$T_\alpha(u) = u \sum_{i=0}^{254} \alpha^i u^{254-i}$$

es el única polinimio de interpolación de Lagrange con la propiedad de que

$$T_\alpha(\beta) = \begin{cases} 1 & , \text{ si } \alpha = \beta, \\ 0 & , \text{ en otros casos.} \end{cases}$$

Si $\alpha = 0$ entonces $T_\alpha(u) = u^{255} + 1$ es el único polinimio de interpolación de Lagrange. Entonces el único polinomio $\phi(u) \in \mathbb{F}[u]$ estará dado por

$$\phi(u) = \sum_{\alpha \in \mathbb{F}} \phi(\alpha) T_\alpha(u).$$

Transformación `ShiftRows`: En esta operación los bytes de la i-ésima fila son desplazados cíclicamente en i posiciones. Algebraicamente esta operación tiene una simple interpretación. Para esto consideremos el elemento $r = r(x, y) \in R$. Entonces **`ShiftRows`** corresponde simplemente a la transformación:

$$r = r(x, y) \mapsto r(xy^3, y),$$

que traslada reemplazando el monomio $x^i y^j$ por el monomio $x^i y^{3i+j}$. La inversa de la transformación **`ShiftRows`** es

$$r = r(x, y) \mapsto r(xy, y),$$

que traslada reemplazando el monomio $x^i y^j$ por el monomio $x^i y^{i+j}$.

Transformación `MixColumns`: En esta transformación cada columna $r_j = \sum_{i=0}^{3} r_{i,j} x^i$ es multiplicada por el elemento λ.

`AddRoundKey`: En este paso la t-ésima clave de ronda es añadida.

El esquema de operaciones es como sigue: En la 'ronda cero' la clave de ronda $k^{(0)}$ es adicionada. En las rondas 1 hasta 9 se realizan las operaciones 'S-box', '**`ShiftRows`**', '**`MixColumns`**' y '**`AddRoundKey`**'. En la décima ronda sólo se emplean las operaciones 'S-box', '**`ShiftRows`**' y '**`AddRoundKey`**'.

10.2.1. AES-192 y AES-256

Hasta ahora, tenemos la descripción del algoritmo Rijndael con tamaño en bloque y clave de 128 bits. Este sistema es conocido como AES-128. En la propuesta original se tiene la posibilidad de variar ambos, el tamaño del bloque y la longitud de la clave secreta.

En el AES como ya sabemos el tamaño del bloque es siempre tomado como 128 bits. En el AES-192 y el AES-256, la longitud de la clave secreta es de 192 y 256 bits respectivamente. Con el propósito de hacer correr éstos presumiblemente más seguros algoritmos, será necesario cambiar el esquema de programación de claves. El el AES-192, 13 elementos $k^{(t)} \in R$, $t = 0, \ldots, 12$ son computados a partir de la clave inicial de 192 bits y el algoritmo debe correr a través de 12 rondas. En el AES-256, son 15 los elementos a ser computados, $k^{(t)} \in R$, $t = 0, \ldots, 14$ derivados de la clave inicial de 256 bits, siendo en este caso necesarias 14 rondas para hacer correr el algoritmo.

10.3. La Estructura de la S-box

Todas las transformaciones son $\mathbb{Z}_2$ lineales, a excepción de la transformación de la S-box. Por consiguiente, un entendimiento de la S-box es determinante.

La permutación ϕ es la composición de las aplicaciones ϕ_1, L y ϕ_3. Describiremos el polinomio de permutación para cada uno de ellos.

El polinimio de permutación para la aplicación ϕ_1 está simplemente dada por $\phi_1(u) = u^{254}$.

La permutación L es una aplicación $\mathbb{Z}_2$-lineal. Por este motivo existe un único polinomio linealizado $\mathcal{L}(u) = \sum_{i=0}^{7} \lambda_i u^{2^i} \in \mathbb{F}[u]$ (sección 3.4) tal que

$$\mathcal{L}(f) = L(f)$$

para todo $f \in \mathbb{F}$. Si $\alpha_1, \ldots, \alpha_8$ es alguna base de $\mathbb{F}$ sobre el campo primo $\mathbb{Z}_2$, entonces es posible calcular los coeficientes $\lambda_0, \lambda_1, \ldots, \alpha_7$ por medio de las ecuaciones lineales:

$$\mathcal{L}(\alpha_j) = \sum_{i=0}^{7} \lambda_i \alpha_j^{2^i} = L(\alpha_j), \quad j = 1, \ldots, 8.$$

Este sistema de ecuaciones lineales puede ser resuelto explícitamente. Para hacer ésto sea $\beta_1, \ldots, \beta_8$ la base dual de $\alpha_1, \ldots, \alpha_8$ caracterizada como ya vimos anteriormente, por el requerimiento:

$$\mathrm{Tr}_{\mathbb{F}/\mathbb{Z}_2}(\alpha_i \beta_j) = \begin{cases} 1 & , \text{ si } i = j, \\ 0 & , \text{ si} i \neq j. \end{cases}$$

Introduciendo las matrices:

$$A = \begin{pmatrix} \alpha_1 & \alpha_1^2 & \alpha_1^4 & \cdots & \alpha_1^{2^7} \\ \alpha_2 & \alpha_2^2 & \alpha_2^4 & \cdots & \alpha_2^{2^7} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_8 & \alpha_8^2 & \alpha_8^4 & \cdots & \alpha_8^{2^7} \end{pmatrix} \quad , B = \begin{pmatrix} \beta_1 & \beta_2 & \cdots & \beta_8 \\ \beta_1^2 & \beta_2^2 & \cdots & \beta_8^2 \\ \beta_1^4 & \beta_2^4 & \cdots & \beta_8^4 \\ \vdots & \vdots & \ddots & \vdots \\ \beta_1^{2^7} & \beta_2^{2^7} & \cdots & \beta_8^{2^7} \end{pmatrix}$$

El hecho de que $\beta_1, \ldots, \beta_8$ sea la base dual de $\alpha_1, \ldots, \alpha_8$ significa simplemente que $AB = I_8$.

Sea S la transformación de cambio de base tal que

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_8 \end{pmatrix} = S \begin{pmatrix} 1 \\ z \\ \vdots \\ z^7 \end{pmatrix}$$

y considérese la matriz

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

misma que describe la aplicación L con respecto a la base polinomial $1, z, z^2, \ldots, z^7$. Entonces tenemos:

LEMA 1. Los coeficientes $\lambda_0, \lambda_1, \ldots, \lambda_7$ de la permutación polinomial $\mathcal{L}(u)$ están dados por:

$$\begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_7 \end{pmatrix} = BSL^T S^{-1} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_8 \end{pmatrix}$$

Demostración. $SL^T S^{-1}$ describe el cambio de bases de la aplicación lineal L con respecto a la base $\alpha_1, \ldots, \alpha_8$. □

Con la intensión de calcular explícitamente los coeficientes $\lambda_0, \lambda_1, \ldots, \lambda_7$ podemos trabajar con la base polinomial $1, z, z^2, \ldots, z^7$ (en tal caso $S = I_8$). Alternativamente podemos trabajar con una base normal. Explicaremos los cálculos para una base normal. Sea

$$\alpha = z^5 + 1 \in \mathbb{F}.$$

Se puede verificar que α es un elemento primitivo de $\mathbb{F}$ y que $\{\alpha_i = \alpha^{2^{i-1}} \mid i = 1, \ldots, 8\}$ constituye una base normal. Tales bases son llamadas bases primitivas normales. α es especial en el sentido de que es el primer elemento de $\mathbb{F}$ con respecto al orden lexicográfico, además de ser primitivo y generador de una bases normal.

Fácilmente se puede calcular la base dual de $\{\alpha_1, \ldots, \alpha_8\}$ a ser $\{\beta_i = \beta^{2^{j-1}} \mid j = 1, \ldots, 8\}$, donde $\beta = z^5 + z^4 + z^2 + 1$. Es bien conocido que la base dual de una base normal es también una base normal. La transformación de cambio de base es calculada en este caso como:

$$S = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Y así con ésto, podemos calcular

$$\begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_7 \end{pmatrix} = BSL^T S^{-1} \begin{pmatrix} \alpha \\ \alpha^2 \\ \vdots \\ \alpha^{2^7} \end{pmatrix} = \begin{pmatrix} z^2 + 1 \\ z^3 + 1 \\ z^7 + z^6 + z^5 + z^4 + z^3 + 1 \\ z^5 + z^2 + 1 \\ z^7 + z^6 + z^5 + z^4 + z^2 \\ 1 \\ z^7 + z^5 + z^4 + z^2 + 1 \\ z^7 + z^3 + z^2 + z + 1 \end{pmatrix}$$

Los elementos λ_i se corresponden con los coeficientes no constantes de ϕ salvo por el orden. Para obtener la exacta forma, necesitamos una descripción polinomial de la permutación ϕ_3. Claramente el polinomio lineal $\phi_3(u) = u + 1 + z + z^5 + z^6 \in \mathbb{F}[u]$ interpola la aplicación afín ϕ_3.

Concatenando las tres aplicaciones polinomiales conseguimos:

$$\phi(u) = (\phi_3 \circ \mathcal{L} \circ \phi_1)(u) = 1 + z + z^5 + z^6 + \mathcal{L}(u^{254}) \mod u^{256} + u.$$

Nótese también que $\mathcal{L}$ tiene a lo más 8 coeficientes no nulos. Reduciendo $\mathcal{L}(u^{254})$ por la relación $u^{256} = u$ no cambiaremos ésto y de aquí la brevedad del polinomio $\phi(u)$.

El hecho de que el polinomio de permutación $\phi(u)$ sea conciso, no implica que el polinomio inverso $\psi(u)$ sea conciso. Para ver ésto nótese que

$$\psi(u) = (\phi_1^{-1} \circ \mathcal{L}^{-1} \circ \phi_3^{-1})(u) \mod u^{256} + u.$$

Como antes, los coeficientes del polinomio $\mathcal{L}^{-1}(u)$ se calculan a través de:

$$BS(L^{-1})^T S^{-1} \begin{pmatrix} \alpha \\ \alpha^2 \\ \vdots \\ \alpha^{2^7} \end{pmatrix}$$

De donde se puede derivar la expresión:

$$\begin{aligned}\mathcal{L}^{-1}(u) = (z^6+z^5+z^3+z^2+z)u^{128} + (z^7+z^6+z^4+z^3+z+1)u^{64} + \\ + (z^6+z^4+z^3+1)u^{32} + (z^6+z^5+z^4+z^3)u^{16} + \\ + (z^6+z^4+z^3+z)u^8 + (z^6+z^5+z^4+z^3+z^2+z+1)u^4 + \\ + (z^7+z^6+z^5+z^4+z^3+z^2+z)u^2 + (z^2+1)u \in \mathbb{F}[u].\end{aligned}$$

Combinando este resultado con la función $\phi_3^{-1}(u)$ obtenemos:

$$\rho(u) = \mathcal{L}^{-1}\phi_3^{-1}(u) = \mathcal{L}^{-1}(u+\phi_3(0)) = \mathcal{L}^{-1}(u) + \mathcal{L}^{-1}(\phi_3(0)) = \mathcal{L}^{-1}(u) + z^2 + 1.$$

Un polinomio de la forma $\rho(u)$ es a veces llamado un polinomio afín, reflejando el hecho de que la aplicación $\mathcal{L}^{-1}\phi_3^{-1}$ es lineal afín sobre $\mathbb{Z}_2$.

Concatenando $\rho(u)$ con el polinomio $\phi_1^{-1}(u) = \phi_1(u) = u^{254}$ obtenemos un polinomio no breve $\psi(u) = \rho(u)^{254} \mod u^{256} + u$. A continuación se presenta el polinomio completo, donde los coeficientes son expresados en términos de del elemento primitivo $\alpha = z^5 + 1 \in \mathbb{F}$.

$$\begin{aligned}\psi(u) = \alpha^{163}u^{254} + \alpha^{76}u^{253} + \alpha^{195}u^{252} + \alpha^{186}u^{251} + \alpha^{234}u^{250} + \alpha^{194}u^{249} + \alpha^{248}u^{248} + \\ + \alpha^{255}u^{247} + \alpha^{196}u^{246} + \alpha^{100}u^{245} + \alpha^{216}u^{244} + \alpha^{212}u^{243} + \alpha^{47}u^{242} + \alpha^{17}u241 + \\ + \alpha^{85}u^{240} + \alpha^{103}u^{239} + \alpha^{201}u^{238} + \alpha^{184}u^{237} + \alpha^{235}u^{236} + \alpha^{215}u^{235} + \alpha^{170}u^{234} + \\ + \alpha^{74}u^{233} + \alpha^{15}u^{232} + \alpha^{2}u^{231} + \alpha^{185}u^{230} + \alpha^{89}u^{229} + \alpha^{26}u^{228} + \alpha^{231}u^{227} + \\ + \alpha^{137}u^{226} + \alpha^{110}u^{225} + \alpha^{230}u^{224} + \alpha^{20}u^{223} + \alpha^{126}u^{222} + \alpha^{35}u^{221} + \alpha^{117}u^{220} + \\ + \alpha^{48}u^{219} + \alpha^{141}u^{218} + \alpha^{56}u^{217} + \alpha^{29}u^{216} + \alpha^{154}u^{215} + \alpha^{207}u^{214} + \alpha^{175}u^{213} + \\ + \alpha^{253}u^{212} + \alpha^{147}u^{211} + \alpha^{5}u^{210} + \alpha^{43}u^{209} + \alpha^{194}u^{208} + \alpha^{242}u^{207} + \alpha^{202}u^{206} + \\ + \alpha^{27}u^{205} + \alpha^{15}u^{204} + \alpha^{164}u^{203} + \alpha^{11}u^{202} + \alpha^{233}u^{201} + \alpha^{56}u^{200} + \alpha^{121}u^{199} + \\ + \alpha^{163}u^{198} + \alpha^{69}u^{197} + \alpha^{113}u^{196} + \alpha^{235}u^{195} + \alpha^{225}u^{194} + \alpha^{152}u^{193} + \alpha^{227}u^{192} + \\ + \alpha^{9}u^{191} + \alpha^{78}u^{190} + \alpha^{234}u^{189} + \alpha^{57}u^{188} + \alpha^{136}u^{187} + \alpha^{115}u^{186} + \alpha^{128}u185 + \\ + \alpha^{57}u^{184} + \alpha^{223}u^{183} + \alpha^{228}u^{182} + \alpha^{110}u^{181} + \alpha^{249}u^{180} + \alpha^{83}u^{179} + \alpha^{55}u^{178} + \\ + \alpha^{55}u^{177} + \alpha^{32}u^{176} + \alpha^{94}u^{175} + \alpha^{71}u^{174} + \alpha^{88}u^{173} + \alpha^{94}u^{172} + \alpha^{45}u^{171} + \\ + \alpha^{218}u^{170} + \alpha^{157}u^{169} + \alpha^{73}u^{168} + \alpha^{209}u^{167} + \alpha^{21}u^{166} + \alpha^{122}u^{165} + \alpha^{127}u^{164} + \\ + \alpha^{206}u^{163} + \alpha^{19}u^{162} + \alpha^{189}u^{161} + \alpha^{89}u^{160} + \alpha^{177}u^{159} + \alpha^{192}u^{158} + \alpha^{211}u^{157} + \\ + \alpha^{99}u^{156} + \alpha^{195}u^{155} + \alpha^{14}u^{154} + \alpha^{172}u^{153} + \alpha^{67}u^{152} + \alpha^{136}u^{151} + \alpha^{6}u^{150} + \\ + \alpha^{122}u^{149} + \alpha^{102}u^{148} + \alpha^{198}u^{147} + \alpha^{14}u^{146} + \alpha^{130}u^{145} + \alpha^{102}u^{144} + \alpha^{129}u^{143} +\end{aligned}$$

$$
\begin{aligned}
&+ \alpha^{246}u^{142} + \alpha^{187}u^{141} + \alpha^{85}u^{140} + \alpha^{181}u^{139} + \alpha^{169}u^{138} + \alpha^{230}u^{137} + \alpha^{21}u^{136} + \\
&+ \alpha^{234}u^{135} + \alpha^{138}u^{134} + \alpha^{104}u^{133} + \alpha^{26}u^{132} + \alpha^{229}u^{131} + \alpha^{177}u^{130} + \alpha^{168}u^{129} + \\
&+ \alpha^{245}u^{128} + \alpha^{13}u^{127} + \alpha^{142}u^{126} + \alpha^{96}u^{125} + \alpha^{240}u^{124} + \alpha^{224}u^{123} + \alpha^{32}u^{122} + \\
&+ \alpha^{228}u^{121} + \alpha^{68}u^{120} + \alpha^{125}u^{119} + \alpha^{147}u^{118} + \alpha^{19}u^{117} + \alpha^{78}u^{116} + \alpha^{51}u^{115} + \\
&+ \alpha^{114}u^{114} + \alpha^{87}u^{113} + \alpha^{120}u^{112} + \alpha^{5}u^{111} + \alpha^{209}u^{110} + \alpha^{51}u^{109} + \alpha^{39}u^{108} + \\
&+ \alpha^{47}u^{107} + \alpha^{109}u^{106} + \alpha^{159}u^{105} + \alpha^{203}u^{104} + \alpha^{202}u^{103} + \alpha^{9}u^{102} + \alpha^{238}u^{101} + \\
&+ \alpha^{44}u^{100} + \alpha^{188}u^{99} + \alpha^{234}u^{98} + \alpha^{59}u^{97} + \alpha^{15}u^{96} + \alpha^{131}u^{95} + \alpha^{173}u^{94} + \alpha^{135}u^{93} + \\
&+ \alpha^{244}u^{92} + \alpha^{216}u^{91} + \alpha^{50}u^{90} + \alpha^{218}u^{89} + \alpha^{250}u^{88} + \alpha^{108}u^{87} + \alpha^{192}u^{86} + \alpha^{45}u^{85} + \\
&+ \alpha^{53}u^{84} + \alpha^{186}u^{83} + \alpha^{92}u^{82} + \alpha^{74}u^{81} + \alpha^{157}u^{80} + \alpha^{172}u^{79} + \alpha^{99}u^{78} + \alpha^{209}u^{77} + \\
&+ \alpha^{236}u^{76} + \alpha^{212}u^{75} + \alpha^{44}u^{74} + \alpha^{209}u^{73} + \alpha^{175}u^{72} + \alpha^{101}u^{71} + \alpha^{41}u^{70} + \alpha^{51}u^{69} + \\
&+ \alpha^{163}u^{68} + \alpha^{183}u^{67} + \alpha^{245}u^{66} + \alpha^{169}u^{65} + \alpha^{58}u^{64} + \alpha^{5}u^{63} + \alpha^{68}u^{62} + \alpha^{63}u^{61} + \\
&+ \alpha^{202}u^{60} + \alpha^{138}u^{59} + \alpha^{204}u^{58} + \alpha^{109}u^{57} + \alpha^{173}u^{56} + \alpha^{214}u^{55} + \alpha^{61}u^{54} + \alpha^{255}u^{53} + \\
&+ \alpha^{185}u^{52} + \alpha^{249}u^{51} + \alpha^{153}u^{50} + \alpha^{143}u^{49} + \alpha^{206}u^{48} + \alpha^{163}u^{47} + \alpha^{43}u^{46} + \alpha^{202}u^{45} + \\
&+ \alpha^{156}u^{44} + \alpha^{70}u^{43} + \alpha^{2}u^{42} + \alpha^{45}u^{41} + \alpha^{81}u^{40} + \alpha^{43}u^{39} + \alpha^{121}u^{38} + \alpha^{90}u^{37} + \alpha^{101}u^{36} + \\
&+ \alpha^{252}u^{35} + \alpha^{42}u^{34} + \alpha^{176}u^{33} + \alpha^{201}u^{32} + \alpha^{22}u^{31} + \alpha^{135}u^{30} + \alpha^{250}u^{29} + \alpha^{176}u^{28} + \alpha^{76}u^{27} + \\
&+ \alpha^{90}u^{26} + \alpha^{247}u^{25} + \alpha^{220}u^{24} + \alpha^{123}u^{23} + \alpha^{76}u^{22} + \alpha u^{21} + \alpha^{180}u^{20} + \alpha^{108}u^{19} + \alpha^{222}u^{18} + \\
&+ \alpha^{54}u^{17} + \alpha^{46}u^{16} + \alpha^{89}u^{15} + \alpha^{240}u^{14} + \alpha^{235}u^{13} + \alpha^{208}u^{12} + \alpha^{194}u^{11} + \alpha^{2}u^{10} + +\alpha^{201}u^{9} + \\
&+ \alpha^{67}u^{8} + \alpha^{247}u^{7} + \alpha^{56}u^{6} + \alpha^{132}u^{5} + \alpha^{16}u^{4} + \alpha^{242}u^{3} + \alpha^{223}u^{2} + \alpha^{243}u + \alpha^{92}
\end{aligned}
$$

Fuera del hecho de que $\psi(u) = \rho(u)^{254} \mod u^{256}+u$, no se observa alguna regularidad en los coeficientes de $\psi(u)$. La tan complicada estructura de la S-box inversa, nos muestra que un ataque algebraico sobre Rijndael que intente resolver de manera recursiva las ecuaciones de desencriptación, se enfrentará con una tarea en extremo difícil.

Capítulo 11

Epílogo

11.1. Conclusiones

Habiendo realizado una exhaustiva exposición del algoritmo AES, haciendo énfasis primordialmente en su estructura matemática fundamental; se ha cumplido con el objetivo central del presente trabajo; pero aun más, se ha llegado a establecer una línea de investigación en el marco del criptoanálisis con respecto a una aparente fragilidad mostrada por el algoritmo aprovechable en función de los fundamentos matemáticos utilizados. Es claro que el diseño del cifrado acorde a la estrategia del rastro amplio, le proporciona un blindaje criptográfico que lo mantiene a salvo de ataques lineales y diferenciales, sin embargo se han logrado avances respecto de un determinado tipo de ataque: el ataque de interpolación, que involucra temas como la interpolación de Lagrange y la Teoría de Ecuaciones para en conjunto intentar vulnerar la encriptación valiéndose de la construcción algebraica inherente al algoritmo.

Por lo demás, solo basta observar el ingenio con el que se ha diseñado el esquema de encriptación, las formas de implementación basadas en representaciones y la aplicación plausible de la teoría algebraica; para tomarlo como punto de referencia en la creación de nuevos métodos criptográficos y criptoanalíticos.

11.2. Recomendaciones

Mi recomendación va en el sentido de brindar mayor fomento a la investigación en ramas afines al tratamiento y la seguridad de datos, que se perfila como una de las ramas de mayor proyección, considerando la creciente e incesante necesidad de contar con conjuntos de datos cada vez más grandes y cuyo manejo de forma segura y plausible será posible gracias los fundamentos de la ciencia matemática, que permitirán la implementación de algoritmos más complejos, mismos que ayudaran en temas de manejo de riesgos, reducción de costos y toma de decisiones. A continuación menciono algo de la problemática que puede

encontrar solución en el desarrollo de herramientas teórico-matemáticas.

Partiendo de que esto no sólo se trata de encriptar y desencriptar mensajes, sino principalmente de resolver problemas del mundo real que requieren seguridad en la información, planteo previamente 4 premisas principales:

- *Confidencialidad:* Eva no debería ser capaz de leer el mensaje que Alicia le envió a Roberto. Los instrumentos principales para llevar a cabo esto son los algoritmos de encriptación y desencriptación.
- *Integridad de los Datos:* Roberto quiere estar seguro de que el mensaje de Alicia no ha sido alterado y es que la transmisión de errores es una posibilidad. Además puede pasar que un adversario sea capaz de interceptar la transmisión y alterar la información antes de que llegue a destino. Existen pues recursos tales como las funciones hash, que proveen métodos de detección de manipulación accidental o malintencionada en los datos.
- *Autenticcación:* Roberto quiere estar seguro que sólo Alicia podría haber enviado el mensaje recibido. Este tópico comprende esquemas de identificación y protocolos de contraseña (en este último caso Roberto sería la computadora). Hay en la actualidad 2 tipos de autenticación que surgen en criptografía: autenticación del ente y autenticación de la fuente de datos. A menudo el término identificación es usado en el contexto de la autenticación del ente, lo cual concierne a probar la identidad de las partes involucradas en la comunicación. La autenticación de la fuente se enfoca en cruzar información acerca del origen de los datos, tal como el creador y el tiempo de creación, con los datos proporcionados.
- *No repudio:* Alicia no puede pretender que no envió el mensaje. El no repudio es particularmente importante en las aplicaciones del comercio electrónico, donde es importante que un consumidor no niegue que dió su autorización para la concreción de una compra. Autenticación y no repudio son conceptos cercanamente relacionados, pero hay una diferencia. En un criptosistema de clave simétrica, Roberto puede estar seguro de que el mensaje viene de Alicia (o de alguien en poder de la clave de Alicia) dado que nadie más podría haber encriptado el mensaje que Roberto desencriptó exitosamente. Por lo tanto, la autenticación es automática. No obstante, el no puede probar ante cualquier otra persona que Alicia fue quien le envió el mensaje, pues hasta podría habérselo enviado el mismo. En consecuencia, el no repudio es esencialmente imposible. En un criptosistema de clave pública, tanto la autenticación como el no repudio pueden ser logrados.

Y he aquí los problemas arriba mencionados:

Firmas Digitales: Por la firma de un documento, el firmante queda vinculado a su contenido y todo lo que éste implique. Se da por sentado que es sumamente difícil para otro individuo reproducir la firma en otro documento, lo cual no

ocurre en los mensajes electrónicos donde las copias son exactas. ¿Cómo prevenir que un adversario corte y pegue la firma de un documento electrónico sobre otro? Esto se lleva a cabo a través de protocolos criptográficos que permiten la verificación de la firma en cualquier documento y de tal forma que el firmante no puede negar su participación.

Identificación: Cuando un usuario inicia sesión en una máquina o inicia un vínculo de comunicación, necesita identificarse. Pero escribir el usuario no es suficiente porque no prueba que el usuario en actividad sea quien dice ser. El uso de una contraseña es habitual en estos casos; pero existen varios métodos de identificación que prueban la identidad de la persona sin revelar su contraseña, como el esquema de identificación de Feige-Fiat-Shamir que está basado en cero-conocimiento del ente.

Establecimiento de Clave: Cuando son grandes cantidades de datos los que se necesitan encriptar, es mejor usar algoritmos de encriptación de clave simétrica. ¿Pero cómo Alicia le entrega la clave secreta a Roberto cuando no tiene oportunidad de encontrarse personalmente con él? Hay muchas maneras de hacer esto. Una es usar criptografía de clave pública. Otro método es el algoritmo de intercambio de clave de Diffie-Hellman. Un enfoque diferente a este problema es contar con un tercer ente confiable que proporcione las claves a Alicia y Roberto. Dos ejemplos son el esquema de generación de clave de Blom y el de Kerberos, el cual es un muy popular protocolo simétrico criptográfico que provee autenticación y seguridad para el intercambio de clave entre usuarios de una red.

Compartimiento del Secreto: Supongamos que se debe otorgar la combinación de la bóveda de un banco a un funcionario; pero por norma no puede ser sólo uno sino al menos dos quienes estén en custodia de esta información. La solución es dividir la combinación entre los designados, de tal forma que al menos dos de ellos estén presentes en el momento de la apertura de la caja fuerte, pero tal división debe hacerse a través de procedimientos de compartimiento de secreto.

Comercio Electrónico: ¿Cómo se puede llevar a cabo transacciones seguras sobre canales abiertos como el internet y cómo se puede proteger la información financiera de comerciantes fraudulentos? Firmas duales es un concepto que plantea una solución a éste tipo de cuestiones.

Dinero Electrónico: Tarjetas de crédito o débito y dispositivos similares son muy convenientes; pero no son anónimos. Claramente una forma de dinero electrónico podría ser útil, al menos para algunas personas. Aunque también debemos tomar en cuenta que las entidades electrónicas pueden ser copiadas. Para el tratamiento de este tipo de problemas existen sistemas de dinero electrónico que suministran anonimato y capturan a falsificadores de dinero virtual.

Manejo Masivo de Datos: El obtener un préstamo bancario o la administración de negocios cuyos servicios sean solicitados por gran cantidad de personas de cualquier parte del mundo y a través de cualquier medio, estará sujeto a la consideración de cantidades masivas de datos, entre los cuales habrán nece-

sariamente datos sensibles. ¿Cómo pueden identificarse tales datos para ser manejados de forma segura en medio de tanto caudal de información en transmisión? La criptografía se desarrollara entonces muy a la par del desarrollo de plataformas de BigData.

Juegos: ¿Cómo jugar por ejemplo al póker entre personas que se encuentran en ubicaciones diferentes? Repartir las cartas desde ya representa un problema. La respuesta al dilema podría provenir de ideas y procedimientos criptográficos.

Bibliografía

[1] *The Design of Rijndael*; *Joan Daemen, Vincent Rijmen*; Springer-Verlag.

[2] *Algebraic Aspects of the Advanced Encryption Standard*; *Carlos Cid, Sean Murphy, Matthew Robshaw*; Springer.

[3] *Finite Fields*; *Rudolf Lidl, Harald Niederreiter*; Cambridge University Press.

[4] *Introduction to Finite Fields and their Applications*; *Rudolf Lidl, Harald Niederreiter*; Cambridge University Press.

[5] *Contemporary Abstract Algebra*; *Joseph A. Gallian*; Brooks/Cole-Ciengage Learning.

[6] *Abstract Algebra*; *David S. Dummit, Richard M. Foote*; John Wiley & Sons Inc.

[7] *Basic Abstract Algebra*; *P. B. Bhattacharya, S. K. Jain, S. R. Nagpaul*; Cambridge University Press.

[8] *Topics in Algebra*; *I. N. Herstein*; John Wiley & Sons Inc.

[9] *Linear Algebra*; *Kenneth Hoffman, Ray Kunze*; Prentice Hall.

[10] *Hiding Data - Selected Topics*; *Rudolf Ahlswede*; Springer.

[11] *Introduction to Cryptography with Coding Theory*; *Wide Trappe, Lawrence C. Washington*; Prentice Hall.

[12] *Applied Algebra: Codes, Ciphers and Discrete Algorithms*; *Darel W. Hardy, Carol L. Walker*; Prentice Hall.

[13] *Linear Groups with an exposition of the Galois Field Theory*; *Leonard Eugene Dickson*; B. G. Teubner.

[14] *Portugaliae Mathematica Vol. 22: A note on Betti-Mathieu Group*; *Leonard Carlitz*; Hermann & Co.

[15] *Journal für die reine und angewandte Mathematik: Polynomials and Linear Transformations over Finite Fields*; *Theresa P. Vaughan*; Walter de Gruyter & Co.

[16] *The Block Cipher Companion*; *Lars R. Knudsen, Matthew J. B. Robshaw*; Springer.

[17] *Using Algebraic Geometry*; *David Cox, John Little, Donal O Shea*; Springer.

[18] *Finite Fields: Theory and Computation*; *Igor E. Shparlinski*; Springer-Science+Business Media. B. V.

[19] *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*; *Federal Information Processing Standards Publication 197*; Publicaciones del National Institute of Standards and Technology NIST-U.S.A.

[20] *AES Proposal: Rijndael*; *Joan Daemen, Vincent Rijmen*; Presentación de propuesta al certamen para la elección del AES.

[21] *A Polynomial Description of the Rijndael Adavanced Encryption Standard*; *Joachim Rosenthal*; Artículo del Departamento de Matemáticas la Universidad de Notre Dame, Indiana U.S.A.

[22] *Rijndael for Algebraists*; *H. W. Lenstra Jr.*; Publicaciones de la Universidad de Berkeley

[23] *Rijndael for Algebraists: An Expanded Version of LenstraŽs Manuscript*; *Hannes Moritz, Wei Zhu*; Trabajos Académicos de la City University of New York Academic.

[24] *Selected Areas in Criptography: A simple Algebraic Representation of Rijndael*; *Niels Ferguson, Richard Schroeppel, Doug Whiting*; Springer.

[25] *In How Many Ways Can You Write Rijndael?*; *Elad Barkan, Eli Biham*; Publicación del Departamento de Ciencia Computacional del Instituto de Tecnología de Israel.

[26] *New Observations on Rijndael*; *Sean Murphy, Matthew Robshaw*; Artículo del Grupo de investigación sobre la Seguridad de la Información de la escuela Royal Holloway de la Universidad de Londres.

[27] *Further Comments on the Struture of Rijndael*; *Sean Murphy and Matthew Robshaw*; Artículo del Grupo de investigación sobre la Seguridad de la Información de la escuela Royal Holloway de la Universidad de Londres.

[28] *Essential Algebraic Structure Within the AES*; *Sean Murphy and Matthew Robshaw*; Artículo del Grupo de investigación sobre la Seguridad de la Información de la escuela Royal Holloway de la Universidad de Londres.

[29] *Answer to "New Observations on Rijndael"*; *Joan Daemen, Vincent Rijmen*; Artículo publicado durante el proceso de selección del nuevo estándar de encriptación AES.

[30] *Finite Fields and Their Applications: Linearized polynomials over finite fields revisited*; *Baofeng Wu, Zhoujun Liu*; Elsevier.

[31] *Finite Fields and Their Applications: A note on linear permutation polynomials*; *Pingzhi Yuan, Xiangneng Zeng*; Elsevier.

[32] *Computation in Finite Fields*; *John Kerl*; Publicación de la Universidad del Estado de Arizona U.S.A. con el auspicio de la Corporación Lockheed Martin.