Criptografia y Seguridad en Redes

v.2015

Seminario #4

Prof.Ing.Miguel SOLINAS

mksolinas@gmail.com

Agenda

- *Introducción*
- *Criptografía histórica*
- *Fundamentos teóricos*
- *Cifra de Bloque Moderna + DES + Modos de Uso*
- *Otras cifras de bloque, 3DES, IDEA, AES*

3DES

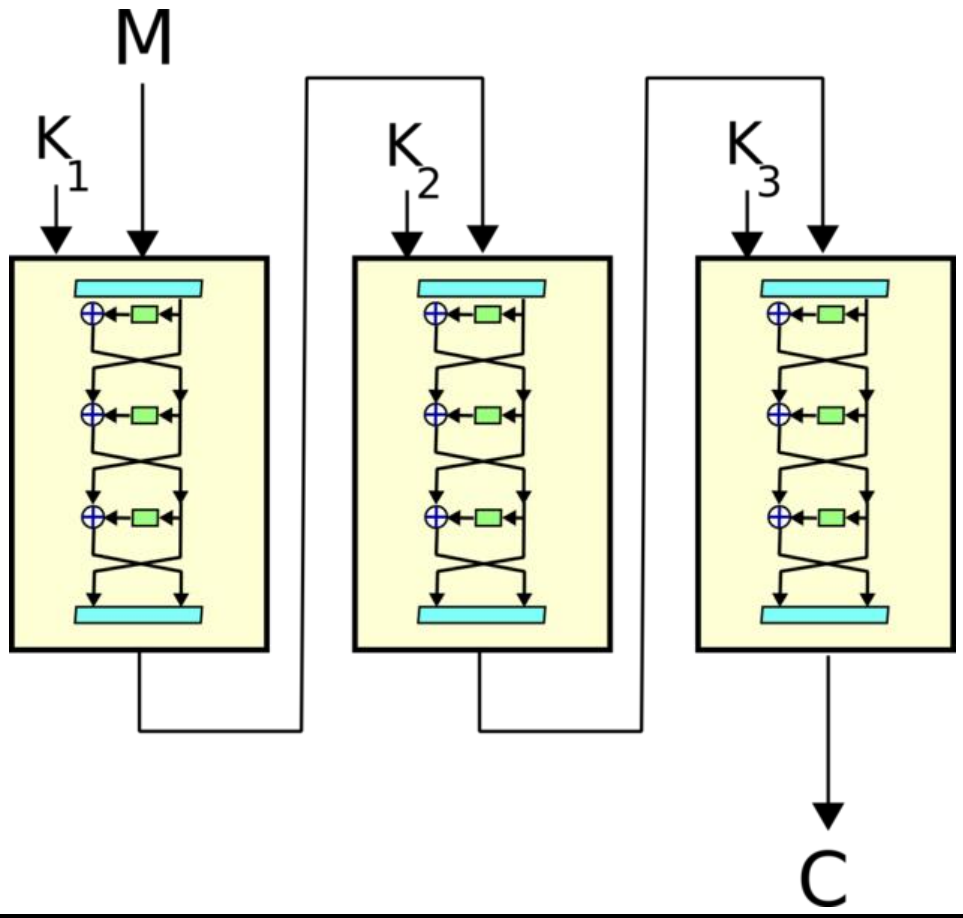
Triple DES

- Se logra un valor efectivo de longitud de clave igual a 2^{2n} bits, es decir $2^{2.56} = 2^{112}$ bits efectivos.
- Es un modelo con dos claves, compatible con DES de clave única si $k_1=k_2$. Es mas eficiente y equivale al cifrado triple con claves k_1,k_2, k_3.
- Fue propuesto por Mayas y Meyer de IBM y se conoce como EDE (Encypt – Decrypt – Encrypt).

Triple DES

$$C=DES_{K3}(DES_{K2}^{-1}(DES_{K1}(M)))$$

La clave resultante es la concatenación de k1 y k2, con una longitud de 112 bits.



IDEA

International Data Encryption Algorithm

IDEA

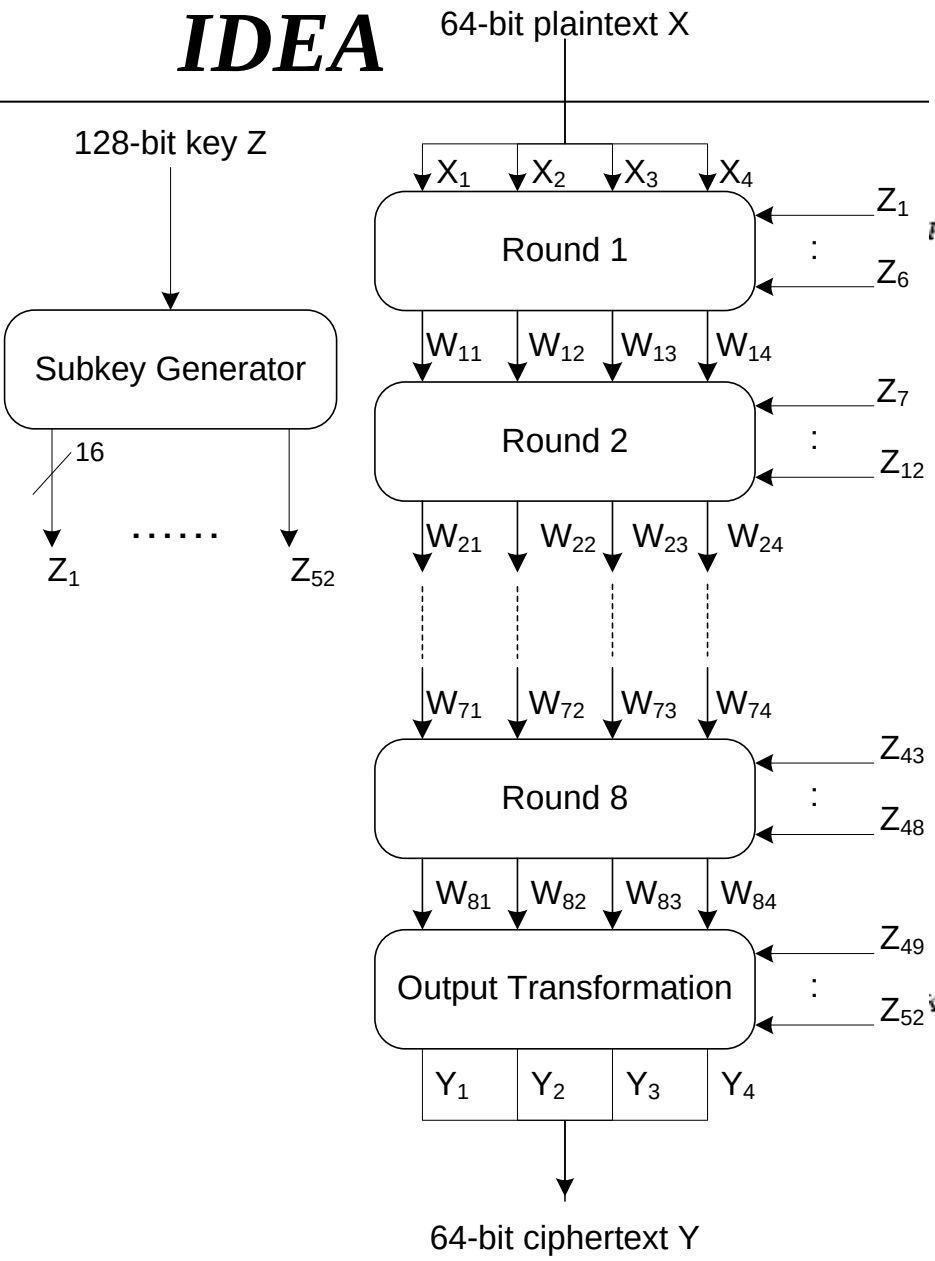
IDEA (International Data Encryption Algorithm)

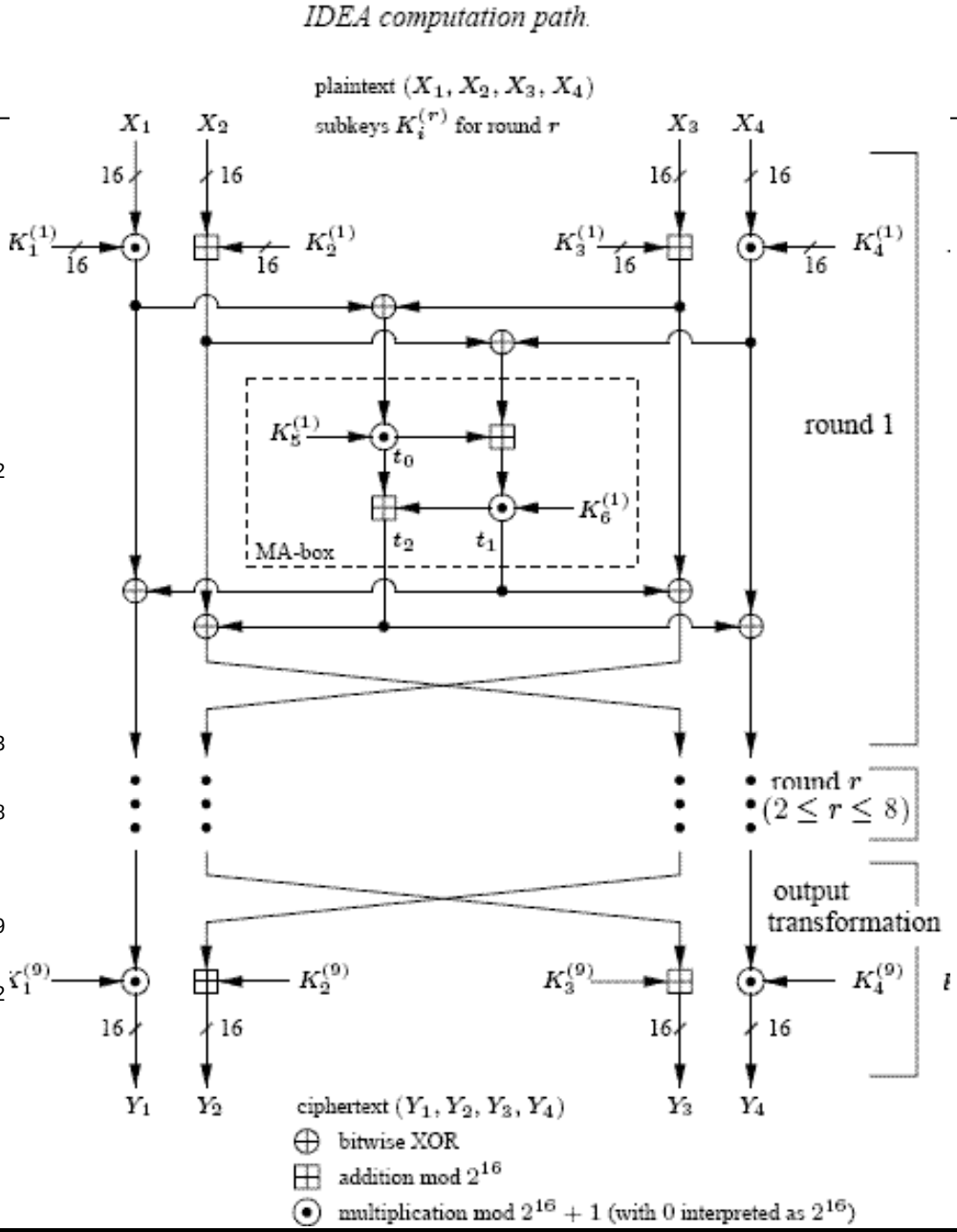
Desarrollado por James Massey & Xuejia Lai en el ETH en Zurich en 1990 y llamado IPES. Revisado en ´91/2 ya como IDEA.

Encripta bloques de 64 bits usando claves de 128 bits, basándose en un mix de operaciones de diferentes (incompatibles) grupos algebraicos:

- XOR,
- suma en mod 2^{16},
- producto en mod $2^{16}+1$.

IDEA


64-bit plaintext X
128-bit key Z
Subkey Generator
16
Z1
Z52
X1 X2 X3 X4
Round 1
Z1
Z6
W11 W12 W13 W14
Round 2
Z7
Z12
W21 W22 W23 W24
W71 W72 W73 W74
Round 8
Z43
Z48
W81 W82 W83 W84
Output Transformation
Z49
Z52
Y1 Y2 Y3 Y4
64-bit ciphertext Y


IDEA computation path.
plaintext (X_1, X_2, X_3, X_4)
subkeys $K_i^{(r)}$ for round r
round 1
MA-box
round r $(2 \le r \le 8)$
output transformation
ciphertext (Y_1, Y_2, Y_3, Y_4)
bitwise XOR
addition mod 2^{16}
multiplication mod $2^{16}+1$ (with 0 interpreted as 2^{16})

IDEA

Planificación de la clave con IDEA

Las sub-claves usadas en las 8 rondas se obtienen del siguiente modo:

- Partir la claves de 128 bits en 8 subclaves de 16 bits.
- Usar 6 claves; rotar el bloque de 128 bits, 25 posiciones a la izquierda y volver a partirlo en 8 subclaves and play again.
- Las subclaves para desencripción se obtienen de un modo mas complejo ya que se debe calcular inversas de las operaciones de suma y multiplicación.

IDEA

Planificación de la clave con IDEA

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

Donde:

K^{-1} es el recíproco de K en mod $2^{16}+1$

-K es el inverso aditivo de K en mod 2^{16}

IDEA

Seguridad de IDEA

- Ataque por fuerza bruta intratable debido al tamaño del espacio de claves.
- Bajo ciertos supuestos puede quebrarse con criptoanálisis diferencial. De todas maneras generalmente se lo considera inmune a este tipo de ataque.
- No se han reportado debilidades frente al criptoanálisis lineal.
- Se han encontrado algunas claves débiles, en la práctica poco utilizadas debiendo evitarse explícitamente.
- Se lo considera uno de los cifrados de bloque mas seguros que existen.

IDEA

Ventajas respecto a DES

- Mayor espacio de claves.
- Todas las operaciones son algebraicas.
- No hay operaciones a nivel de bit, esto facilita programación en alto nivel.
- Más eficiente que algoritmos tipo Feistel, porque a cada vuelta se modifican todos los bits de bloque y no solamente la mitad.
- Mas simple, seguro y de igual velocidad que DES.
- Utiliza todos los modos de operación definidos para DES (ECB, CBC, CFB, OFB).

IDEA

Implementación

- Fácilmente implementable en software y hardware.
- Se utilizó en PGP v2.0 y es una opción en OpenPGP.
- Kits de desarrollo de software para la implementación del algoritmo en la bibliografía
- ¿ El uso de IDEA requiere de una licencia ?

AES

Advanced Encryption Standard

AES

En 1997, el NIST decidió realizar un concurso para escoger un nuevo algoritmo de cifrado capaz de proteger información sensible durante el siglo XXI.

El 2 de enero de 1997 el NIST anunció su intención de desarrollar AES, con la ayuda de la industria y de la comunidad criptográfica.
El 12 de septiembre de ese año se hizo la convocatoria formal en la que se indicaban varias condiciones para los algoritmos que se presentaran:

- Ser de dominio público, disponible para todo el mundo.
- Ser un algoritmo de cifrado simétrico y soportar bloques de, como mínimo, 128 bits.
- Las claves de cifrado podrían ser de 128, 192 y 256 bits.
- Ser implementable tanto en hardware como en software.

AES

El 20 de agosto de 1998 el NIST anunció los 15 algoritmos admitidos en la primera conferencia AES:

1. **CAST-256** (*Entrust Technologies, Inc.*)
2. **CRYPTON** (*Future Systems, Inc.*)
3. **DEAL** (*Richard Outerbridge, Lars Knudsen*)
4. **DFC** (*CNRS – Centre National pour la Recherche Scientifique – Ecole Normale Superieure*)
5. **E2** (*NTT – Nippon Telegraph and Telephone Corporation*)
6. **FROG** (*TecApro International, S.A.*)
7. **HPC** (*Rich Schroeppel*)
8. **LOKI97** (*Lawrie Brown, Josef Pieprzyk, Jennifer Seberry*)
9. **MAGENTA** (*Deutsche Telekom AG*)
10. **MARS** (*IBM*)
11. **RC6** (*RSA Laboratories*)
12. **RIJNDAEL** (*John Daemen, Vincent Rijmen*)
13. **SAFER+** (*Cylink Corporation*)
14. **SERPENT** (*Ross Anderson, Eli Biham, Lars Knudsen*)
15. **TWOFISH** (*B.Schneier, J.Kelsey, D.Whiting, D.Wagner, C.Hall, N.Ferguson*)

AES

La $2_{da.}$ conferencia AES tuvo lugar en marzo de 1999. Se admitieron comentarios hasta abril y en agosto de 1999 el NIST decidió los 5 finalistas:

1. **CAST-256** (*Entrust Technologies, Inc.*)
2. **CRYPTON** (*Future Systems, Inc.*)
3. **DEAL** (*Richard Outerbridge, Lars Knudsen*)
4. **DFC** (*CNRS – Centre National pour la Recherche Scientifique – Ecole Normale Superieure*)
5. **E2** (*NTT – Nippon Telegraph and Telephone Corporation*)
6. **FROG** (*TecApro International, S.A.*)
7. **HPC** (*Rich Schroeppel*)
8. **LOKI97** (*Lawrie Brown, Josef Pieprzyk, Jennifer Seberry*)
9. **MAGENTA** (*Deutsche Telekom AG*)
10. **MARS** (*IBM*)
11. **RC6** (*RSA Laboratories*)
12. **RIJNDAEL** (John Daemen, Vincent Rijmen)
13. **SAFER+** (*Cylink Corporation*)
14. **SERPENT** (Ross Anderson, Eli Biham, Lars Knudsen)
15. **TWOFISH** (B.Schneier, J.Kelsey, D.Whiting, D.Wagner, C.Hall, N.Ferguson)

AES

En mayo de 2000 finalizó el periodo público de análisis. El NIST estudió toda la información disponible para decidir cual sería el algoritmo ganador. El 2 de octubre de 2000 se votó cual sería el algoritmo que finalmente ganaría el concurso. El resultado fue el siguiente:

1. **MARS** : 13 votos
2. **RC6** : 23 votos
3. **RIJNDAEL** : 86 votos
4. **SERPENT** : 59 votos
5. **TWOFISH** : 31 votos

El algoritmo Rijndael ganó el concurso y en noviembre de 2001 se publicó FIPS 197 donde se asumía oficialmente.

AES

AES no posee estructura de red de Feistel..!!

En su lugar se ha definido cada ronda como una composición de cuatro funciones invertibles diferentes, formando tres capas, diseñadas para proporcionar resistencia frente a criptoanálisis lineal y diferencial.

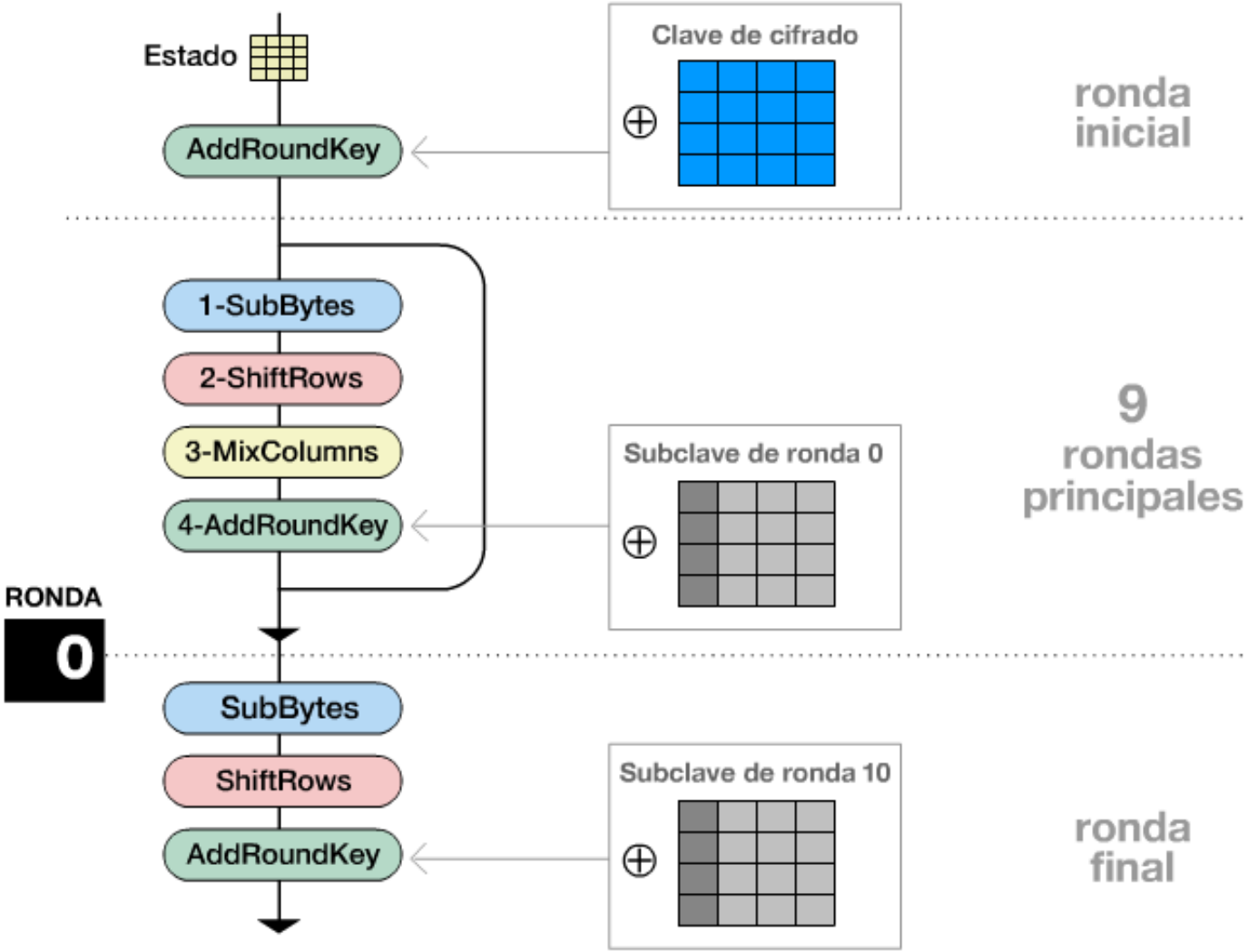
Cada una de las funciones tiene un propósito preciso:

SubByte: Capa no lineal que consiste en la aplicación paralela de sBox con propiedades óptimas de no linealidad.

ShiftRows y **MixColumns**: Esta mezcla lineal permite obtener un alto nivel de difusión a lo largo de varias rondas.

AddRoundKey: La capa de adición de clave es un simple XOR entre el estado intermedio y la sub-clave correspondiente a cada ronda.

AES


Estado
AddRoundKey
Clave de cifrado
ronda inicial
1-SubBytes
2-ShiftRows
3-MixColumns
4-AddRoundKey
Subclave de ronda 0
9 rondas principales
RONDA 0
SubBytes
ShiftRows
AddRoundKey
Subclave de ronda 10
ronda final

AES

El algoritmo Rijndael opera a nivel de bytes, interpretándolos como elementos de un cuerpo de Galois GF(2^8), y a nivel de registros de 32 bits, considerándolos como polinomios de grado menor que 4 con coeficientes que son a su vez polinomios en GF(2^8).

Los Galois Fields (GF) se utilizan en criptografía porque en ellos existe un inverso aditivo y multiplicativo que permite cifrar y descifrar en el mismo cuerpo Z_k, eliminando así los problemas de redondeo o truncamiento de valores si tales operaciones de cifrado y descifrado se hubiesen realizado en aritmética real.

AES

Interesa utilizar una aritmética en módulo “p” sobre polinomios de grado “m”, siendo “p” un número primo. Este GF queda representado como: $GF(p^m)$, en donde los elementos de $GF(p^m)$ se representan como polinomios con coeficientes en Z_p de grado menor que m, es decir:

$GF(p^m) = \{ b_0 + b_1x + b_2x^2 + \dots b_{m-1}x^{m-1} \}$; $b_0, b_1, b_2 \dots b_{m-1}$ pertenecen a Z_p

Cada elemento de $GF(p^m)$ es un resto módulo p(x), donde p(x) es un polinomio irreducible de grado “m”, esto es, que no puede ser factorizado en polinomios de grado menor que “m”.

En el caso del algoritmo serán interesante los campos del tipo $GF(2^m)$ puesto que los coeficientes en este caso serán los restos del módulo 2, es decir, 0 y 1, lo que permite una representación binaria. Por lo tanto, cada elemento del campo se representa con “m” bits y el número de elementos será 2^m.

Operación de SUMA de byte, GF(2^8)

Suma de dos polinomios de tamaño byte expresados dentro de GF(2^8) :

$A = 57_{16} = 0101\ 01112 = x^6 + x^4 + x^2 + x + 1$

$B = 83_{16} = 1000\ 00112 = x^7 + x + 1$

Sumando:

$A+B = (x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) \bmod 2$

$A+B = (x^7 + x^6 + x^4 + x^2 + 2x + 2) \bmod 2 = x^7 + x^6 + x^4 + x^2 = 1101\ 0100 = D4_{16}$

Lo mismo se obtiene utilizando la operación XOR:

$0101\ 0111$ XOR $1000\ 0011 = 1101\ 0100 = D4_{16}$

$\{57_{16}\}$ XOR $\{83_{16}\} = \{D4_{16}\}$

Operación de MULTIPLICACIÓN de byte, GF(2^8)

Para la multiplicación de polinomios se realiza modulo con un polinomio irreducible de grado 8. Este polinomio irreducible se representa por m(x) (es irreducible porque sus únicos divisores son el 1 y el mismo polinomio). El polinomio irreducible utilizado en el algoritmo es : $m(x) = x^8+x^4+x^3+x+1$

Ejercicio: Multiplicar A por B

$A = 57_{16} = 0101\ 0111_2 = x^6 + x^4 + x^2 + x + 1$

$B = 83_{16} = 1000\ 0011_2 = x^7 + x + 1$

$A \otimes B = (x^6 + x^4 + x^2 + x + 1) \times (x^7 + x + 1) \bmod 2$

$A \otimes B = 1100\ 0001 = C1_{16}$

Operación xTIME, GF(2^8)

Multiplicación de un polinomio b(x) por x, dentro de GF(2^8); se utiliza para la reducción de exponente un polinomio primitivo irreducible :

$$b(x)=b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0$$

$$b(x) \cdot x = b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x^2 + b_0x^1$$

Una vez tenemos este resultado se debe realizar la reducción modulo m(x).

Si b_7=0 el resultado es el mismo polinomio.

Si b_7=1, m(x) debe anular el valor de x^8.

Operación xTIME, GF(2^8)

En general, para este tipo de multiplicación los autores definen una función denominada “xtime” que simplifica la multiplicación de un polinomio por potencias de x, gracias a que la función xtime se puede ejecutar de forma reiterativa.

La función “xtime” consiste en aplicar un desplazamiento a la izquierda al valor que representa el polinomio y una operación XOR con el valor 0x11B (0x11B = 000100011011= m(x) = x8+x4+x3+x+1) cuando el resultado de la multiplicación debe ser reducido módulo m(x).

Operación xTIME, GF(2^8)

Esta función se puede programar fácilmente de la siguiente manera:

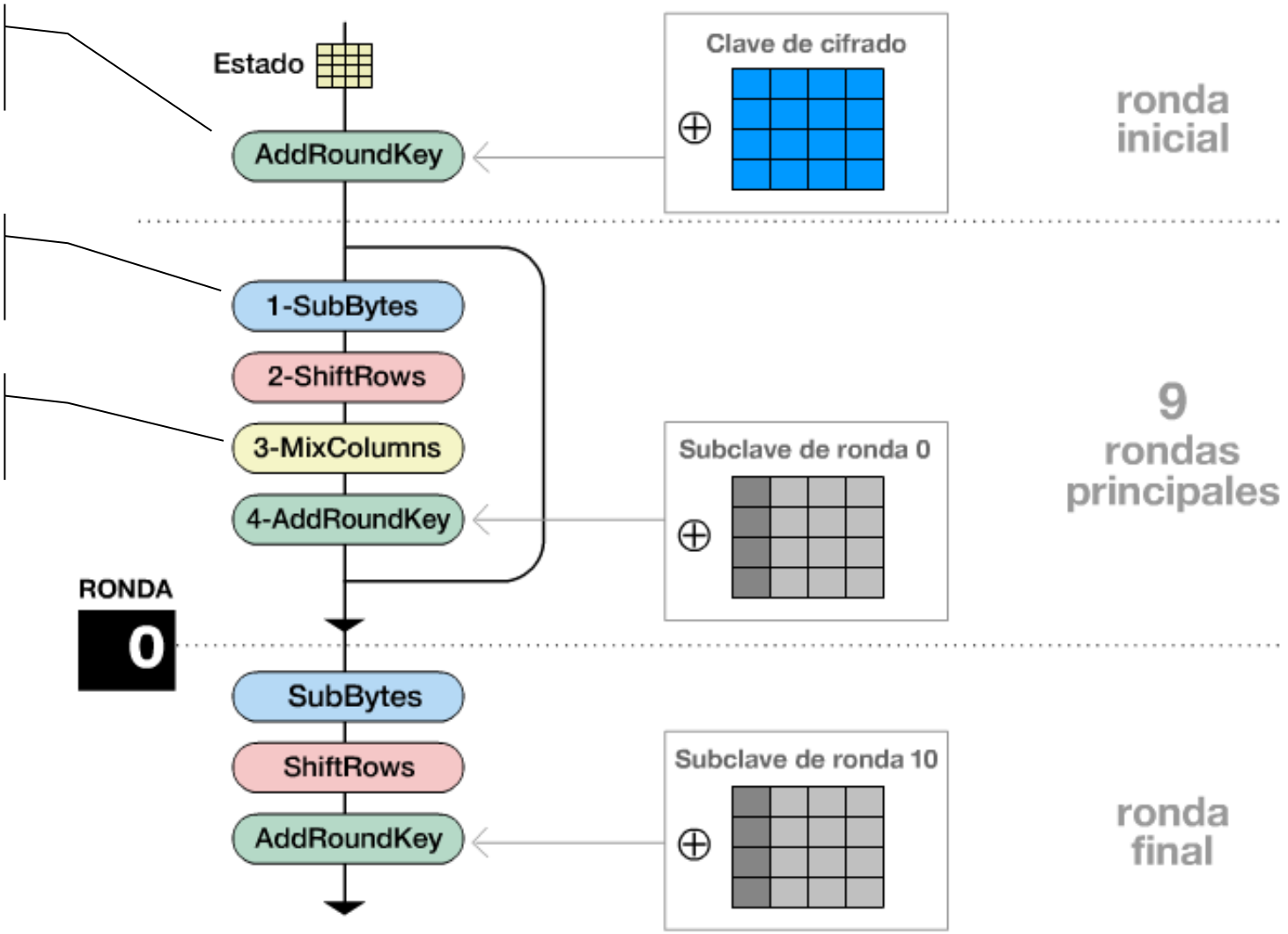
```
int xtime (int valor) {
      valor = valor << 1;
      if (valor&0x100)
            valor^=0x11B;
      return valor;
}
```

AES

Suma (XOR) GF(2^8)

Multiplicación GF(2^8) módulo m(x)

Multiplicación GF(2^8) módulo x^4-1



¿ Animación ?

(http://www.formaestudio.com/rijndaelinspector/)

¿ Es AES seguro ?

(http://aes.cryptosystem.net/)

Diseño e implementación de un prototipo en FPGA

(http://www.criptored.upm.es/guiateoria/gt_m595a.htm)


Welcome to the
UTAH DATA CENTER
IF YOU HAVE
NOTHING TO HIDE
YOU HAVE NOTHING TO FEAR

Otras cifra de bloque

Camellia

(https://info.isl.ntt.co.jp/crypt/eng/camellia/)

Otras…

(http://embeddedsw.net/Cipher_Reference_Home.html/)


GRACIAS!