

4.5 Decryption

* Al no estar basado en una red Feistel, todas las capas se deben invertir: la susg. dos byte se convierte en invención de susl. por bytes y así. Además, las operaciones inversas son similares a las Operaciones y el orden de las llaves se invierte:

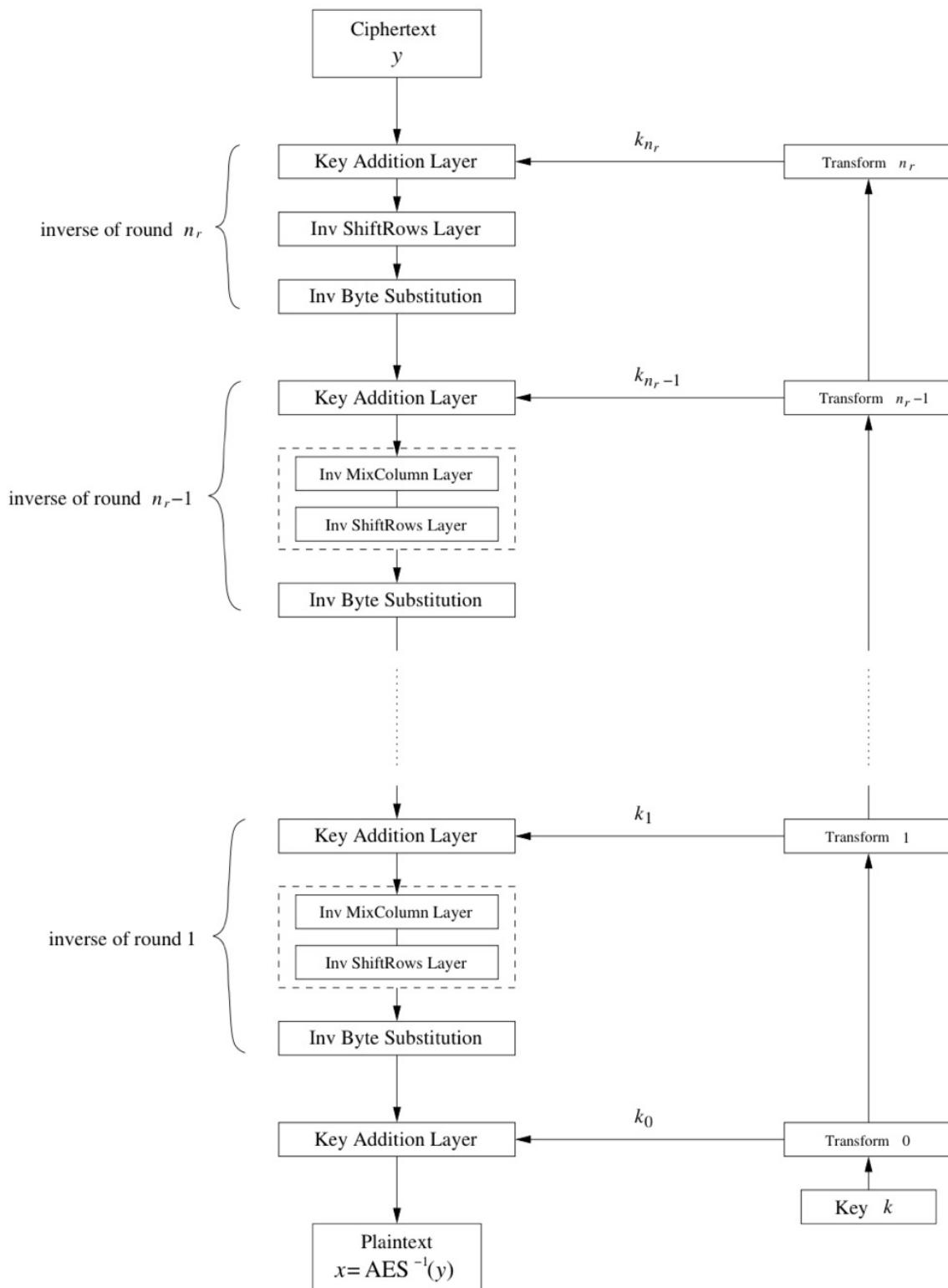


Fig. 4.8 AES decryption block diagram

$\text{XOR} \longleftrightarrow$ su propia inversa.

La capa de adición de llave en el modo de descifrado es la misma: consiste en una fila de casilleros XOR.

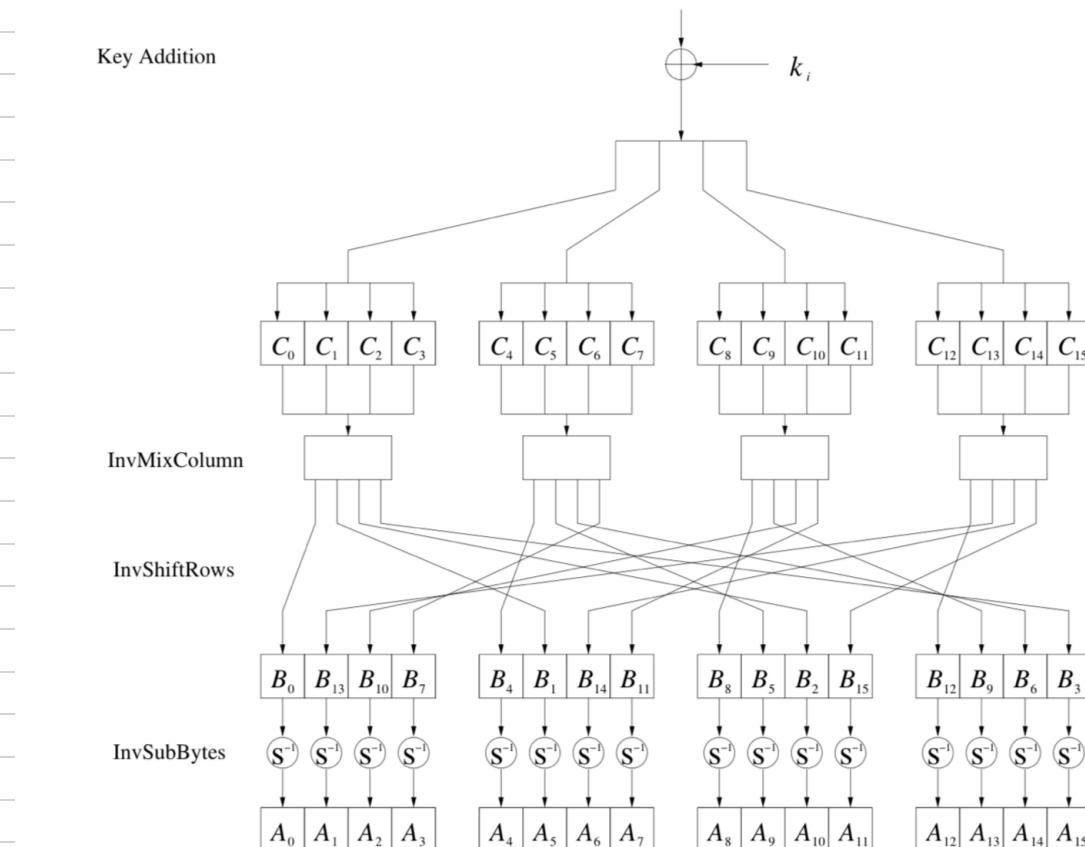


Fig. 4.9 AES decryption round function $1, 2, \dots, n_r - 1$

Capa inversa de MixColumn

Después de la adición de llave, se aplica la inversa de MixColumns a la matriz de estado (excepto en la primera ronda). Se utiliza una matriz inversa de 4×4 convoluciones constantes. La operación se realiza en $\text{GF}(2^8)$. Cada columna de la matriz estado se multiplica por este matriz inversa usando la operación XOR bit a bit.

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix}$$

$B_i, C_i \in \text{GF}(2^8)$

$\underbrace{\hspace{2cm}}$

todos los elem. están en $\text{GF}(2^8)$

La notación para los constantes es hexadecimal:

$$0B = (0B)_{\text{hex}} = (00001011)_2 = x^3 + x + 1$$

Capa inversa de desplazamiento de filas

Para realizar esta operación, las filas de la matriz de estado se desplazan en dirección opuesta. La primera no cambia pero las demás se reordenan para restaurar su posición original.

Si el input esté dado por la matriz estado $B = (B_0, B_1, \dots, B_{15})$:

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

el output de la inversa sería:

B_0	B_4	B_8	B_{12}
B_{13}	B_1	B_5	B_9
B_{10}	B_{14}	B_2	B_6
B_7	B_{11}	B_{15}	B_3

no shift

← three positions left shift

← two positions left shift

← one position left shift

Capa inversa de sustitución de B_{xy}

Durante el descifrado, se usa una S-Box inversa (es una función biyectiva).

$$A_i = S^{-1}(B_i) = S^{-1}(S(A_i)), A_i, B_i \text{ elementos de la matriz de estado}$$

Table 4.4 Inverse AES S-Box: Substitution values in hexadecimal notation for input byte (xy)

	y															
x	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	0B	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

La inversión se realiza en dos pasos:

(1) Se calcula la trans. de fin inversa, tratando cada byte como elemento en $GF(2^8)$

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} \equiv \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \text{ mod } 2,$$

T resultados

la representación binaria de B_i

- 2) Se invertir la operación en el campo utilizando el polinomio de reducción $P(x) = x^8 + x^4 + x^3 + x + 1$
 Polinomio $A_i = (A_i^{-1})^{(1)}$, la op. inversa es "reversa" al calcular la inversa de mueso (A_i) con la otra $A_i = (B_i^{-1})^{(1)} \in GF(2^8)$
 con el pol. de reducción. El $O \rightarrow O$. El vector $A_i \cdot (a_7, \dots, a_0)$ [el elem. $a_7x^7 + \dots + a_0x + a_0$] es el resultado de la sustitución
 $A_i \cdot S^{-1}(B_i)$

El resultado de la sust. inversa es el valor original de la matriz de estado ante del cifrado

Ejercicio de clave para el descifrado

La primera ronda de descifrado requiere la última subclave, la segunda la penúltima y así, ant. las subclaves se usan en el orden inverso.

En la práctica, se calculan previamente todo el esquema de llaves y se almacenan las 11, 13 o 15 subclaves.

4.6 Implementación en Software y Hardware

Software

- AES fue diseñado para ser más eficiente en Software que DES.
- No es óptima implementarlo en un procesador moderno de 32 o 64 bits debido a que las operaciones en byte a byte.
- Los creadores de Rijndael propusieron la técnica/usa de T-Boxes para mejorar la eficiencia.
- Las T-boxes agrupan las funciones en tablas de búsqueda de 256 entradas de 32 bits. Esto permite calcular cada ronda con 16 accesos a los tablas, lo que logra velocidades de 16 Gbit/s en procesadores de 64 bits.

Hardware

- AES requiere más recursos que DES.
- Existen implementaciones de alta velocidad que superan los 10 Gbit/s en chips dedicados.
- Con la paralelización se puede aumentar el rendimiento
- Si se compara con otros algoritmos el AES es extremadamente rápido.

4.7 Discusión y Lecturas Adicionales

- * AES ha sido estudiado intensamente desde los años 90 y no se conocen ataques más eficientes que la fuerza bruta.
- * Si existen estudios de ataques algebraicos pero no han resultado viables en la práctica.
- * Implementaciones eficientes utilizan las T-boxes.
- * Intel introdujo instrucciones AES en 2008 para mejorar la velocidad en hardware.

4.8 Conclusiones

- * AES es un cifrado moderno que soporta llaves de 128, 192, 256 bits y ofrece seguridad contra ataques de fuerza bruta.
- * No se han encontrado ataques prácticos mejores que la fuerza bruta.
- * No utiliza redes Feistel, sino aritmética en campo de Galois para lograr una fuerte difusión y confusión.
- * Es parte de estándares como IPsec y TLS y es el algoritmo obligatorio para aplicaciones gubernamentales.
- * Su eficiencia en software y hardware lo convirtió/convierte en el algoritmo dominante para el cifrado en años recientes.

$S \sqrt{2} \text{ bits} \Rightarrow 2^{32} \text{ lenguajes}$