Blockverschlüsselung und AES

Proseminar/Seminar
Kryptographie und Datensicherheit

SoSe 2009 – Universität Potsdam

ein Vortrag von

Linda Tschepe

Übersicht

- Allgemeines
- SPNs (Substitutions- Permutations- Netzwerke)
- Lineare Kryptoanalyse
- Differentielle Kryptoanalyse
- DES (der Data Encryption Standard)
- AES (der Advanced Encryption Standard)

Allgemeines - Blockchiffre

Was ist eine Blockchiffre?

Bei der Blockverschlüsselung werden immer Blöcke einer festen Länge verschlüsselt.

Moderne Blockchiffren sind meist Produktchiffren, welche Substitutions- und Permutationschiffren kombinieren.

Allgemeines – eine typische iterierte Chiffre

$w^0 \leftarrow x$

$w^1 \leftarrow g(w^0, K^1)$

$w^2 \leftarrow g(w^1, K^2)$

:

:

$w^{N-1} \leftarrow g(w^{N-2}, K^{N-1})$

$w^N \leftarrow g(w^{N-2}, K^N)$

$y \leftarrow w^N$

- N – Anzahl der Runden
- K – binärer Schlüssel
- $(K^1,\ldots,K^N)$ – Liste der Rundenschlüssel
- w^r – Zustand des Textes
  w^0 – Klartext (x)
  w^N – Chiffretext (y)
- g – Rundenfunktion :
  $g(w^{r-1}, K^r) = w^r$

Allgemeines – eine typische iterierte Chiffre

$$w^N \leftarrow y$$

$$w^{N-1} \leftarrow g^{-1}(w^N, K^N)$$

$$\vdots$$

$$\vdots$$

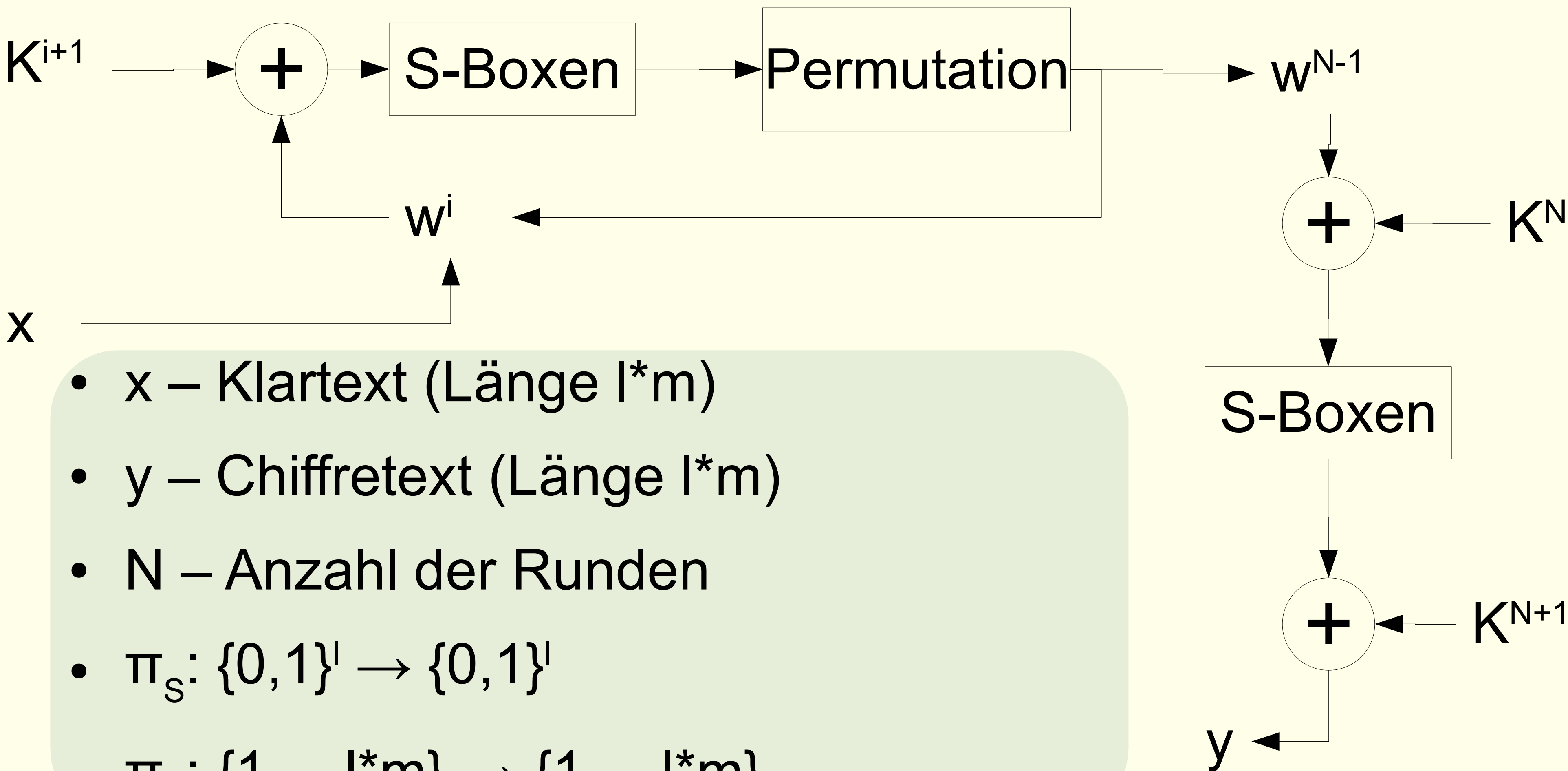
$$w^0 \leftarrow g^{-1}(w^1, K^1)$$

$$x \leftarrow w^0$$

g muss injektiv sein !!!

- N – Anzahl der Runden
- K – binärer Schlüssel
- $(K^1, \ldots, K^N)$ – Liste der Rundenschlüssel
- w^r – Zustand des Textes
  w^0 – Klartext (x)
  w^N – Chiffretext (y)
- g – Rundenfunktion :
  $g(w^{r-1}, K^r) = w^r$

Substitutions-Permutations-Netzwerke



- x – Klartext (Länge l*m)
- y – Chiffretext (Länge l*m)
- N – Anzahl der Runden
- $\pi_S: \{0,1\}^l \rightarrow \{0,1\}^l$
- $\pi_P: \{1,\ldots,l*m\} \rightarrow \{1,\ldots,l*m\}$

SPN - Beispiel

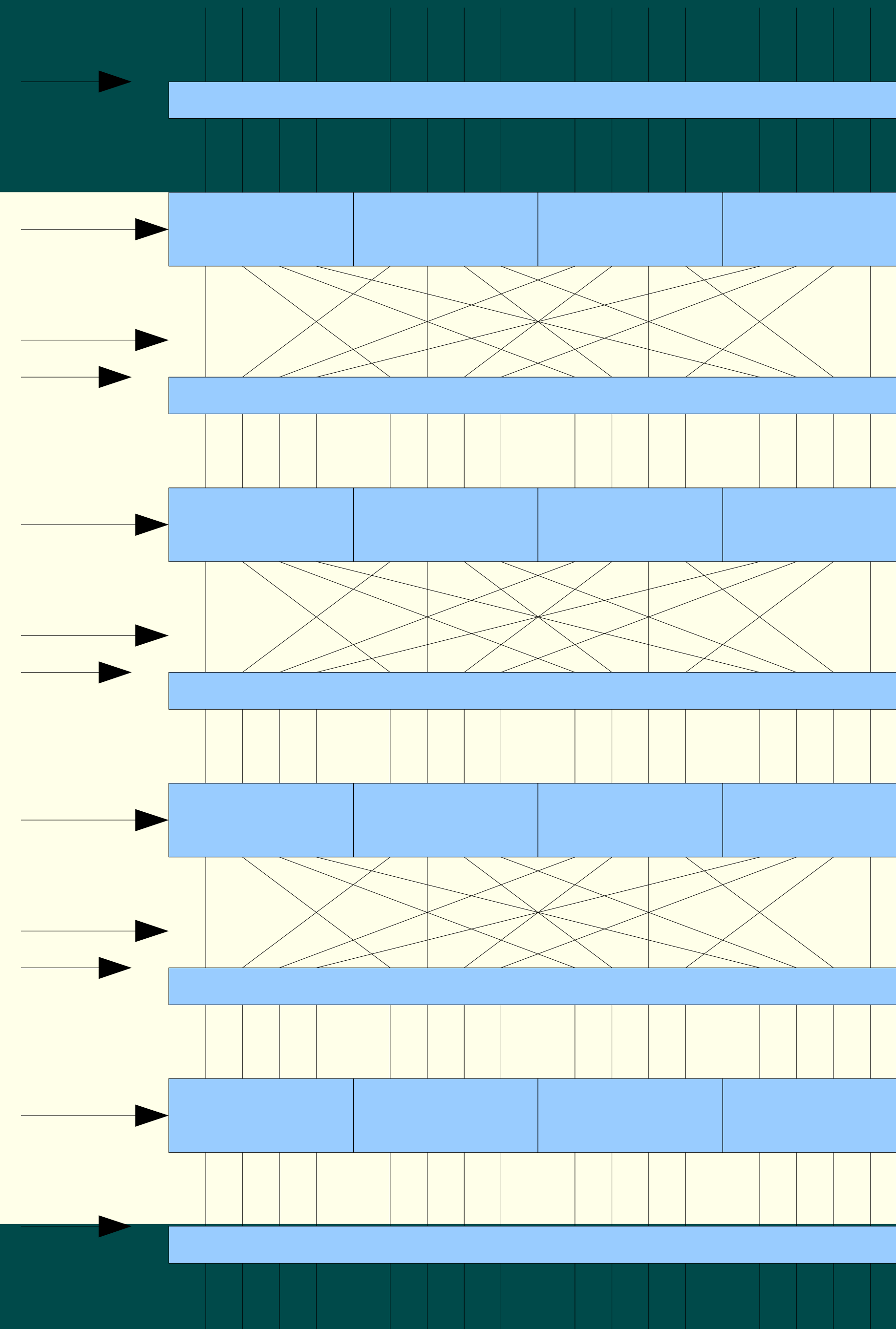
π_S :

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

π_P :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

K^1: 0011 1010 1001 0100
K^2: 1010 1001 0100 1101
K^3: 1001 0100 1101 0110
K^4: 0100 1101 0110 0011
K^5: 1101 0110 0011 1111



Lineare Kryptoanalyse

- 1993 von Mitsuru Matsui publiziert
- kann prinzipiell auf jede iterierte Chiffre angewandt werden
- fällt unter die Kategorie „Known-Plaintext-Attacke“
- Lineare Chiffren sind relativ leicht zu entziffern
  - ➔ Umgehen der nicht-linearen S-Boxen
- Idee: Approximation der Chiffrierfunktion durch eine lineare Abbildung

Lineare Kryptoanalyse – Piling-up Lemma

Seien $X_1, \dots, X_n$ unabhängige Zufallsvariablen, welche Werte der Menge $\{0,1\}$ annehmen, und $\epsilon_1, \dots, \epsilon_n$ die zugehörigen Bias-Werte, so gilt:

$$P(X_1 \text{ xor } \dots \text{ xor } X_n = 0) = \tfrac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

$P(X_i=0) = p_i$

$P(X_i=1) = 1-p_i$

$\epsilon_i = p_i - \tfrac{1}{2}$

$P(X_i=0) = \tfrac{1}{2} + \epsilon_i$

$P(X_i=1) = \tfrac{1}{2} - \epsilon_i$

Lineare Kryptoanalyse – 1. Teil

Approximieren der S-Boxen durch eine lineare Abbildung:

4 Eingangsvariablen: X_1, X_2, X_3, X_4
4 Ausgangsvariablen: Y_1, Y_2, Y_3, Y_4

Untersuchen aller Abbildungen der Form ($a_i, b_i \in \{0,1\}$):

$$a_1X_1 \text{ xor } a_2X_2 \text{ xor } a_3X_3 \text{ xor } a_4X_4 = b_1Y_1 \text{ xor } b_2Y_2 \text{ xor } b_3Y_3 \text{ xor } b_4Y_4$$

$a_1a_2a_3a_4$ in Hexadezimaldarstellung = Inputsumme (u)
$b_1b_2b_3b_4$ in Hexadezimaldarstellung = Outputsumme (v)

Lineare Kryptoanalyse – 1. Teil

0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Lineare Approximations-Tabelle:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
1	8	8	6	6	8	8	6	14	10	10	8	8	10	10	8	8
2	8	8	6	6	8	8	6	6	8	8	10	10	8	8	2	10
3	8	8	8	8	8	8	8	8	10	2	6	6	10	10	6	6
4	8	10	8	6	6	4	6	8	8	6	8	10	10	4	10	8
5	8	6	6	8	6	8	12	10	6	8	4	10	8	6	6	8
6	8	10	6	12	10	8	8	10	8	6	10	12	6	8	8	6
7	8	6	8	10	10	4	10	8	6	8	10	8	12	10	8	10
8	8	8	8	8	8	8	8	8	6	10	10	6	10	6	6	2
9	8	8	6	6	8	8	6	6	4	8	6	10	8	12	10	6
A	8	12	6	10	4	8	10	6	10	10	8	8	10	10	8	8
B	8	12	8	4	12	8	12	8	8	8	8	8	8	8	8	8
C	8	6	12	6	6	8	10	8	10	8	10	12	8	10	8	6
D	8	10	10	8	6	12	8	10	4	6	10	8	10	8	8	10
E	8	10	10	8	6	4	8	10	6	8	8	6	4	10	6	8
F	8	6	4	6	6	8	10	8	8	6	12	6	6	8	10	8

Lineare Kryptoanalyse – 1. Teil

Die lineare Approximations-Tabelle zeigt die Anzahl der möglichen Kombinationen für $X_1X_2X_3X_4$, für die die entsprechende Gleichung wahr ist.

z.B: Inputsumme: B, Outputsumme: 1 -> 12

$\epsilon(B1) = (12 - 8)/16 = 1/4$

$P(B1) = 1/2 + \epsilon(B1) = 3/4$

Je größer $|\epsilon(xy)|$ ist desto, besser ist die Approximation zu verwenden.
(Bei $\epsilon(xy) = 1/2$ ist eine perfekte Repräsentation gefunden.)

Lineare Kryptoanalyse – 2. Teil

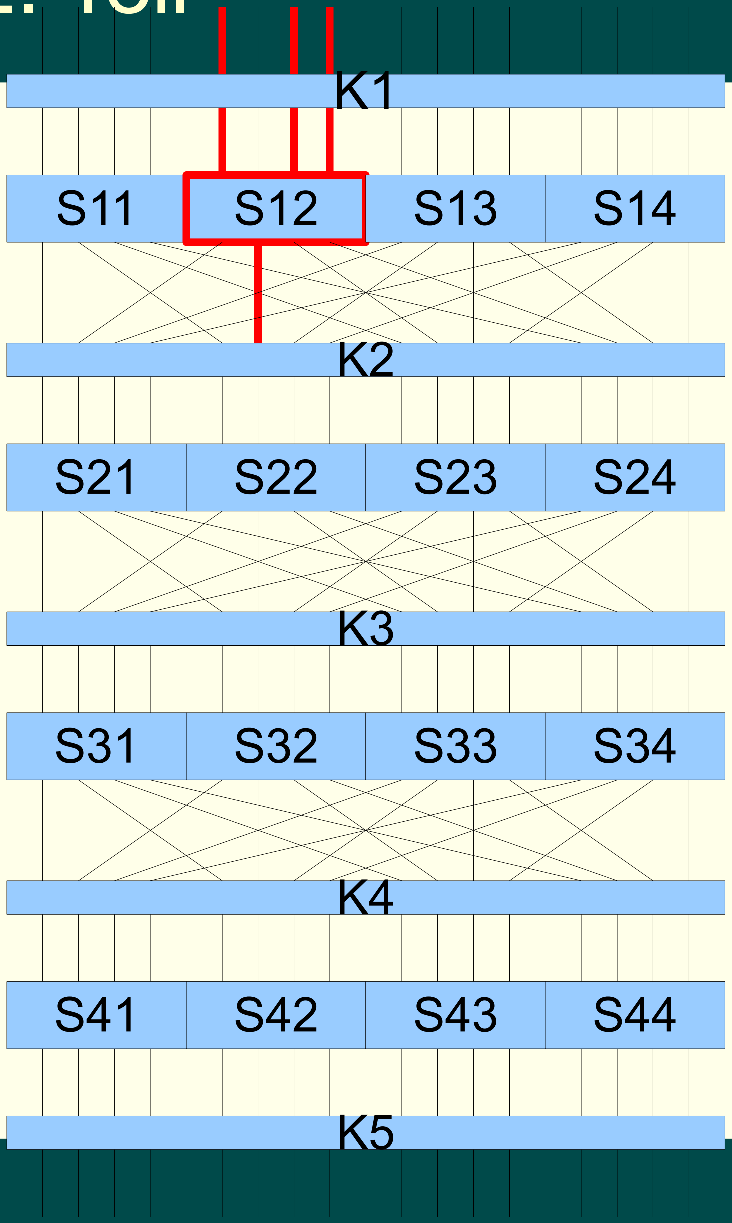
Zusammensetzen der einzelnen Approximationen zu einem Ganzen:

Input S-Box $S_{i,j} = U_{i,k}$ (k є {1, ..., 16})

Output S-Box $S_{i,j} = V_{i,k}$ (k є{1,...,16})

Approximation $S_{1,2}$: (Wahrscheinlichkeit ¾) $U_{1,5}$ xor $U_{1,7}$ xor $U_{1,8}$ = $V_{1,6}$

X_5 xor X_7 xor X_8 xor $K_{1,5}$ xor $K_{1,7}$ xor $K_{1,8}$ = $V_{1,6}$



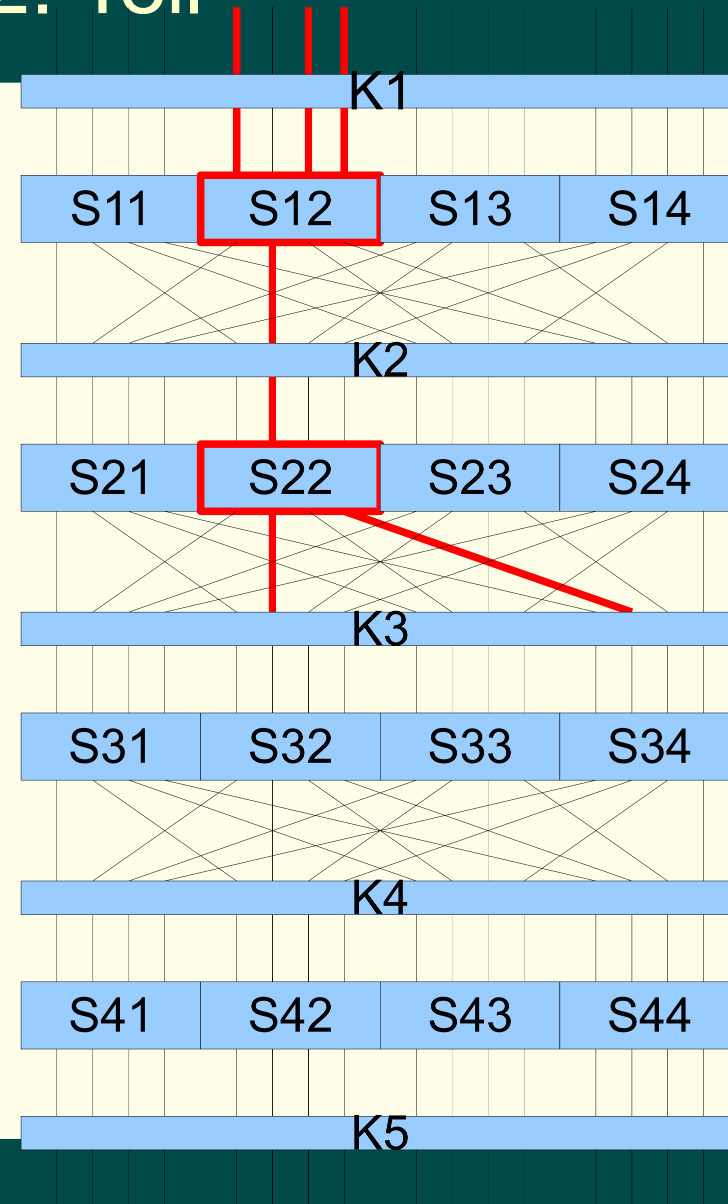
Lineare Kryptoanalyse – 2. Teil

Approximation $S_{2,2}$: (Wahrscheinlichkeit ¼) $U_{2,6} = V_{2,6}$ xor $V_{2,8}$

$U_{2,6} = V_{1,6}$ xor $K_{2,6}$

X_5 xor X_7 xor X_8 xor $K_{1,5}$ xor $K_{1,7}$ xor $K_{1,8}$ xor $K_{2,6}$ = $V_{2,6}$ xor $V_{2,8}$

(mit Wahrscheinlichkeit: $\frac{1}{2} + 2(\frac{3}{4} - \frac{1}{2})(\frac{1}{4} - \frac{1}{2}) = 3/8$) (Piling-Up Lemma)



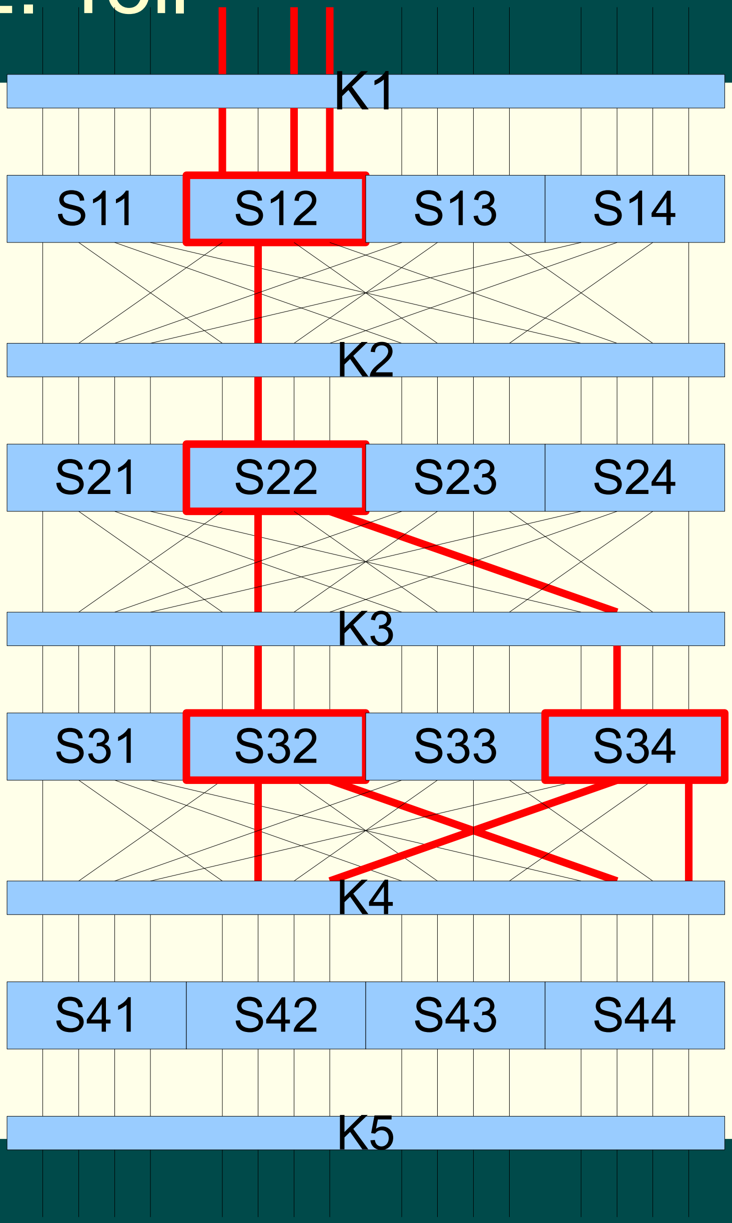
Lineare Kryptoanalyse – 2. Teil

Approximation $S_{3,2}$: (Wahrscheinlichkeit ¼) $U_{3,6} = V_{3,6}$ xor $V_{3,8}$

Approximation $S_{3,4}$: (Wahrscheinlichkeit ¼) $U_{3,14} = V_{3,14}$ xor $V_{3,16}$

$U_{3,6} = V_{2,6}$ xor $K_{3,6}$, $U_{3,14} = V_{2,8}$ xor $K_{3,14}$

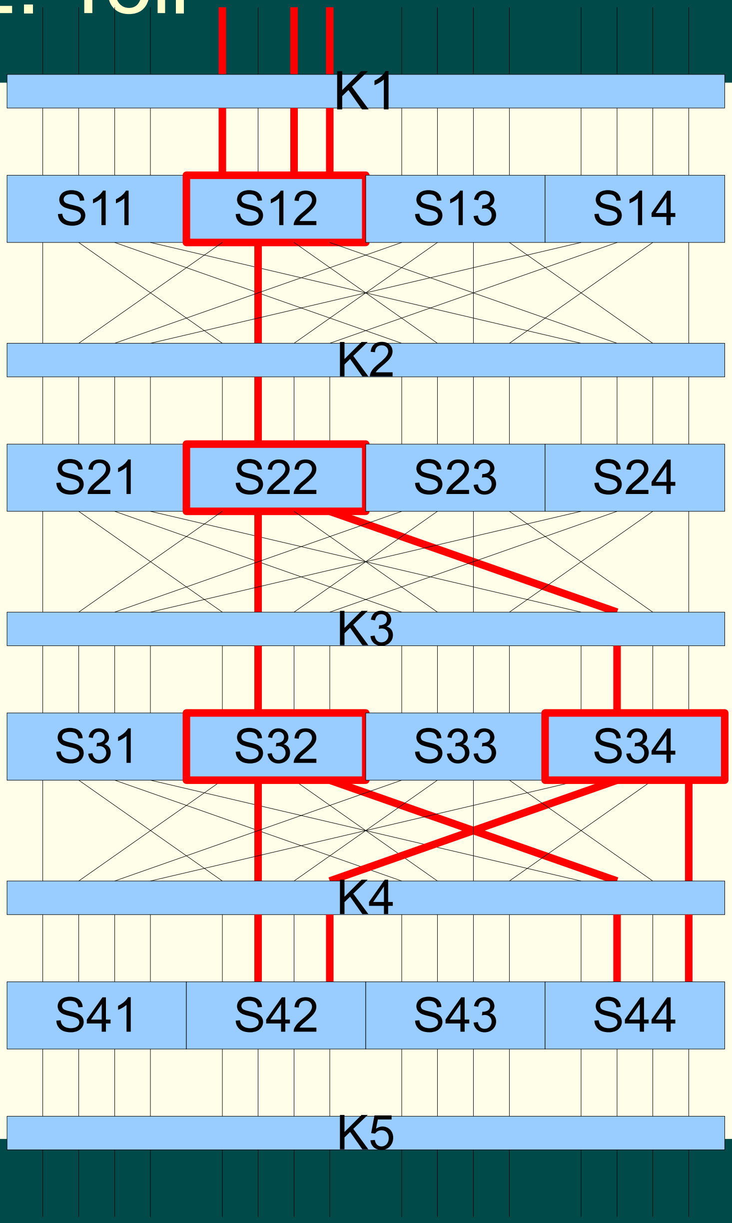
X_5 xor X_7 xor X_8 xor $K_{1,5}$ xor $K_{1,7}$ xor $K_{1,8}$ xor $K_{2,6}$ xor $K_{3,6}$ xor $K_{3,14}$ = $V_{3,6}$ xor $V_{3,8}$ xor $V_{3,14}$ xor $V_{3,16}$ (Wahrscheinlichkeit:15/32)



Lineare Kryptoanalyse – 2. Teil

X_5 xor X_7 xor X_8 xor
$K_{1,5}$ xor $K_{1,7}$ xor $K_{1,8}$ xor $K_{2,6}$ xor $K_{3,6}$ xor
$K_{3,14}$ xor $K_{4,6}$ xor $K_{4,8}$ xor $k_{4,14}$ xor $K_{4,16}$
= $U_{4,6}$ xor $U_{4,8}$ xor $U_{4,14}$ xor $U_{4,16}$

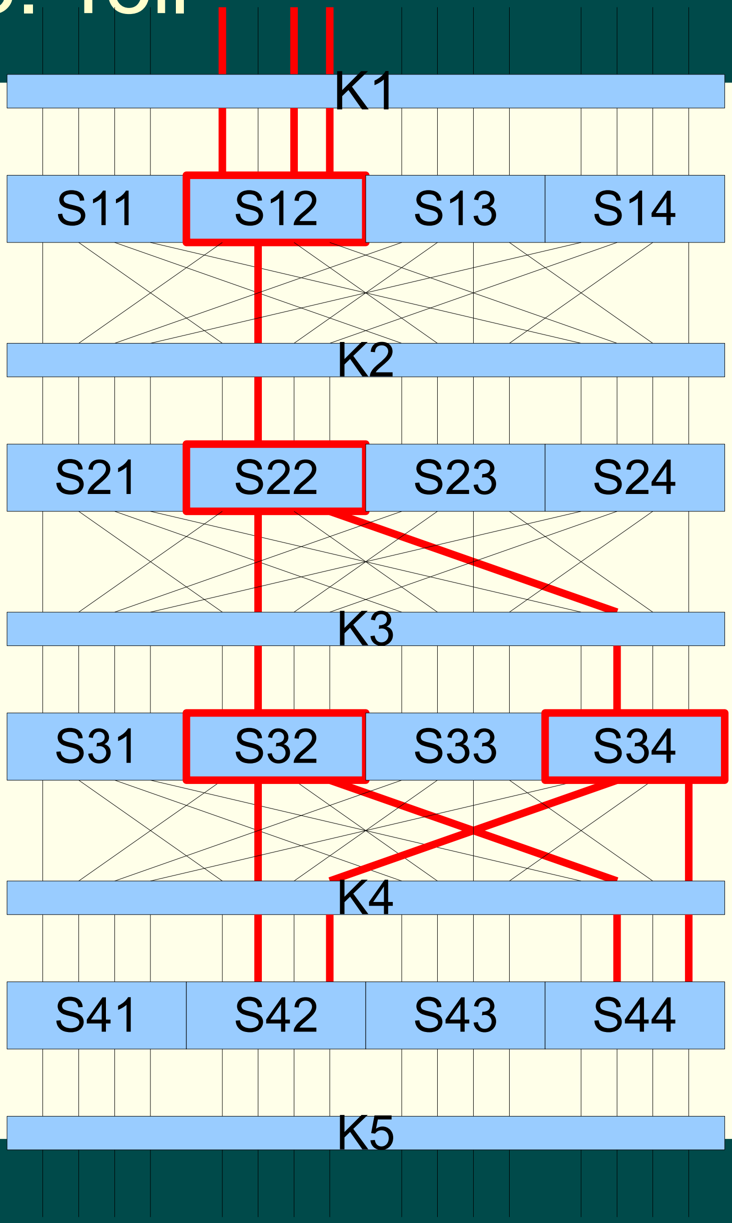
($K_{1,5}$ xor $K_{1,7}$ xor $K_{1,8}$ xor $K_{2,6}$ xor $K_{3,6}$ xor
$K_{3,14}$ xor $K_{4,6}$ xor $K_{4,8}$ xor $k_{4,14}$ xor $K_{4,16}$)
hat festen Wert 1 oder 0
→ X_5 xor X_7 xor X_8 xor $U_{4,6}$ xor $U_{4,8}$ xor
$U_{4,14}$ xor $U_{4,16}$ = 0 oder 1



Lineare Kryptoanalyse – 3. Teil

Berechnen einzelner Bits des letzten Teilschlüssels:

- y xor K_5
- Rückwärts durch S-Box
- Wiederholen für jedes Klar-, Chiffretextpaar (Werte Zählen)
- Der Wert dessen Zähler am weitesten entfernt von (Anzahl Paare)/2 ist, also den größten Bias-Wert hat, ist wahrscheinlich der richtige



Differentielle Kryptoanalyse

- 1990 von Biham & Shamir publiziert
- Fällt in Kategorie „Chosen-Plaintext-Attacke“
- Ist der linearen Kryptoanalyse recht ähnlich
- Hauptunterschied: die differentielle Kryptoanalyse vergleicht den Xor-Wert zweier Inputs mit dem der zugehörigen Outputs
- Also den Unterschied der Eingabe $\Delta X = X'$ xor X'' mit dem Unterschied der Ausgabe $\Delta Y = Y'$ xor Y''

Differentielle Kryptoanalyse

- $\Delta X = X' \text{ xor } X''$, $\Delta Y = Y' \text{ xor } Y''$
- Bei einer ideal zufällig erscheinenden Chiffre ist die Wahrscheinlichkeit, dass auf ein ΔX ein bestimmtes ΔY folgt ½^(n) (n- Anzahl der Bits von X)
- Der Angreifer sucht nach einem ΔX, für welches ein ΔY mit hoher Wahrscheinlichkeit erscheint

Differentielle Kryptoanalyse

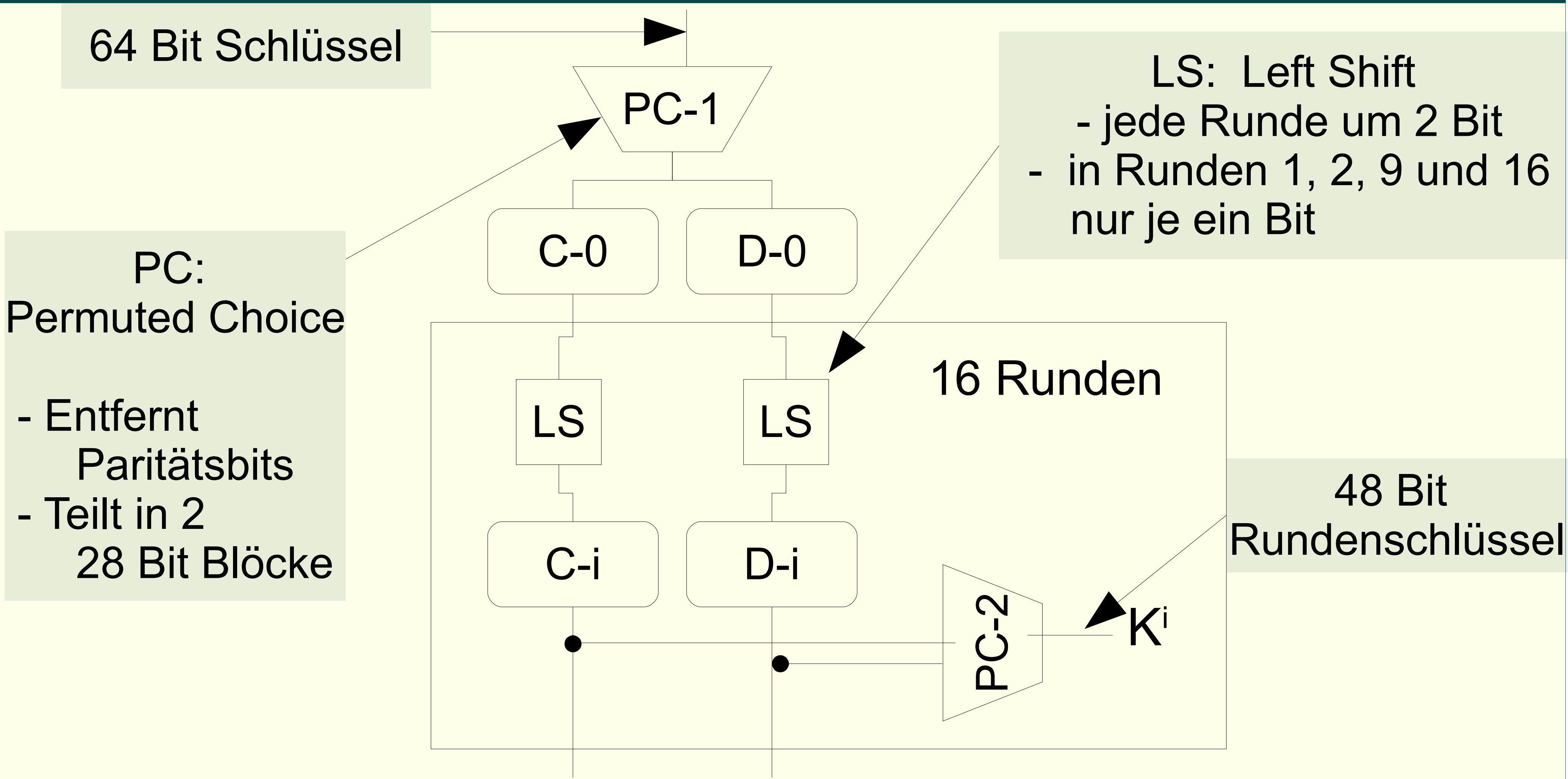
- Differentielle Charakteristiken sind Sequenzen von Ein- und Ausgabedifferenzen von Runden, wobei die Ausgabedifferenz der einen Runde der Eingabedifferenz der nächsten entspricht
- Suche noch möglichst wahrscheinlichen differentiellen Charakteristiken bis in die letzte Runde
  - ➔ Rückschlüsse auf einzelne Bits des letzten Teilschlüssels

Data Encryption Standard

- Vorgängersystem: Lucifer (IBM)
- Entwickelt von IBM mit Hilfe der NSA (National Security Agency)
- 1977 in den USA als Standard festgelegt
- 2000 durch AES abgelöst

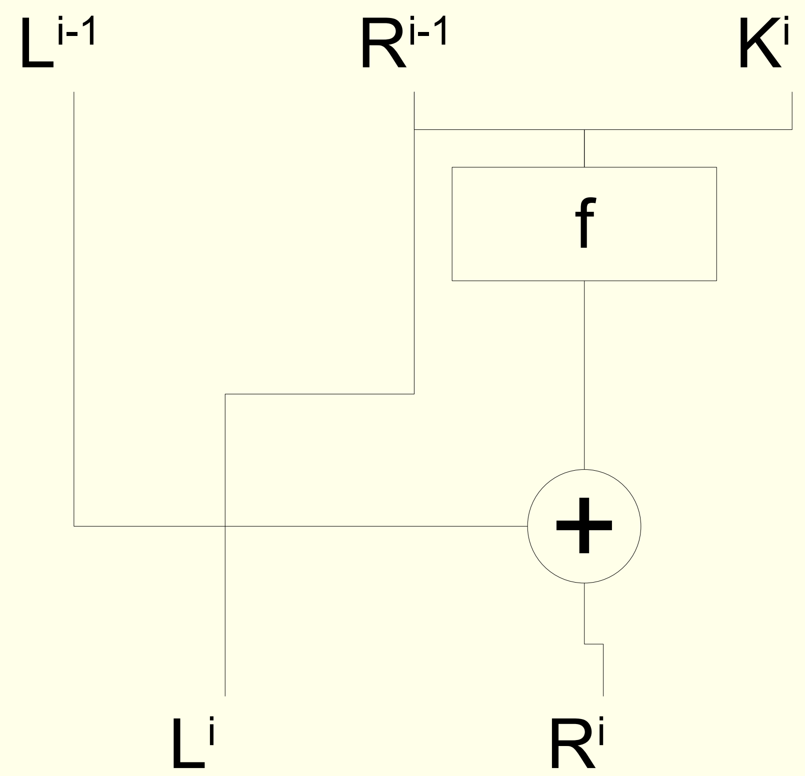
- Modifizierte Feistel - Chiffre
  - Iterierte Blockchiffre
  - Rundenanzahl: 16
  - Blocklänge: 64 Bit
  - Schlüssellänge: 64 Bit (56 Bit + 8 Paritätsbits)
    - → Schlüsselraum: 2^{56}

DES – Berechnen der Rundenschlüssel



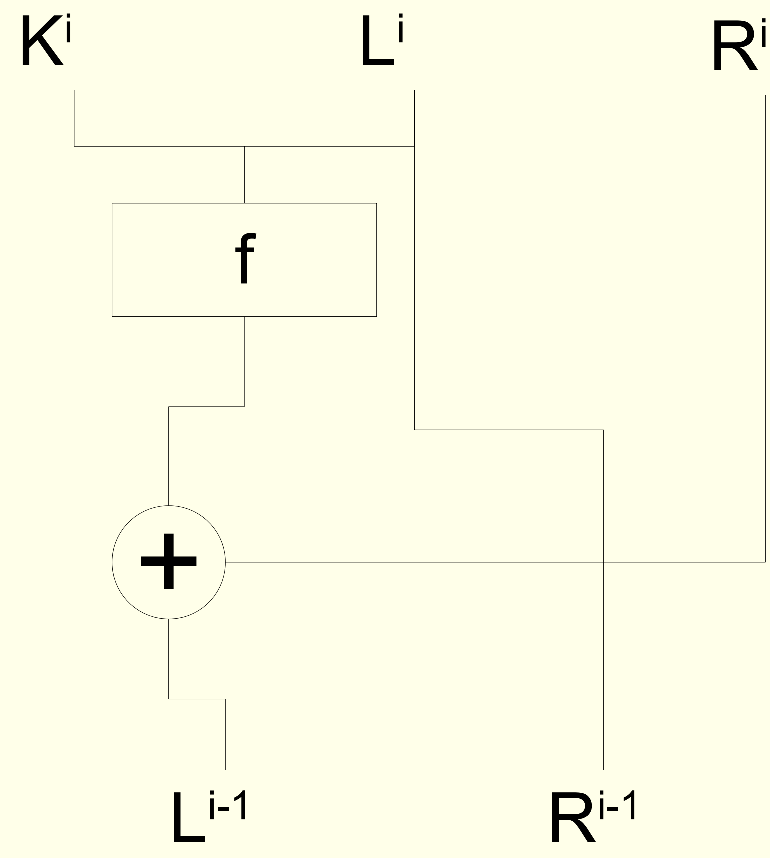
DES - Verschlüsselung

1. Unterteilung des Chiffretextes in 2 Blöcke gleicher Länge (L^0,R^0)
2. Rundenfunktion: $g(L^{i-1},R^{i-1},K^i) = (L^i,R^i)$
   - $L^i = R^{i-1}$
   - $R^i = L^{i-1}$ xor $f(R^{i-1},K^i)$
3. Vertauschen der Blöcke

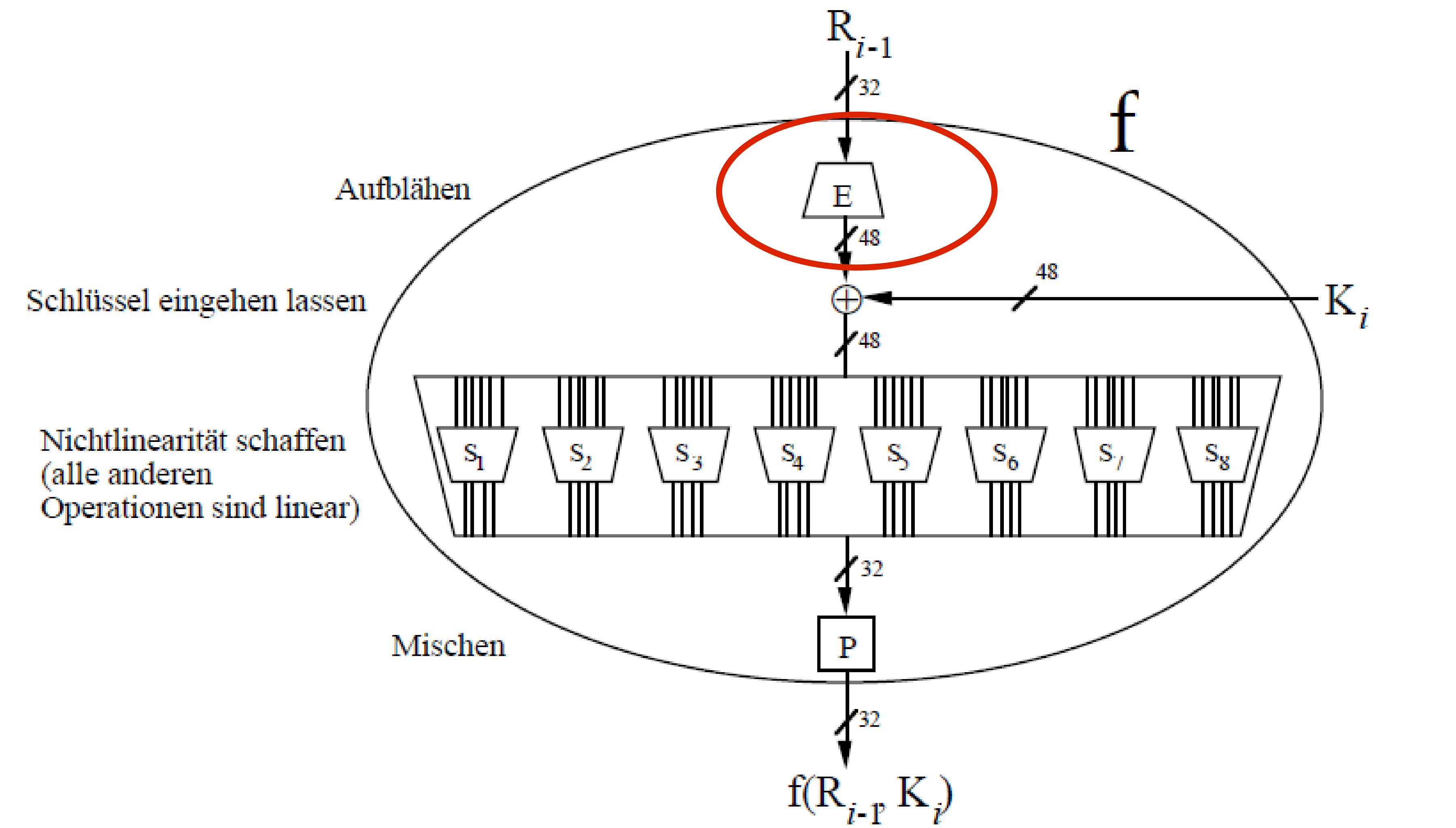


DES – Entschlüsselung

1. Zerteilen des Chiffretextes und Vertauschen der Hälften
2. Rundenfunktion:
   - $R^{i-1} = L^{i}$
   - $L^{i-1} = R^{i} \text{ xor } f(L^{i},K^{i})$



DES – die Verschlüsselungsfunktion f

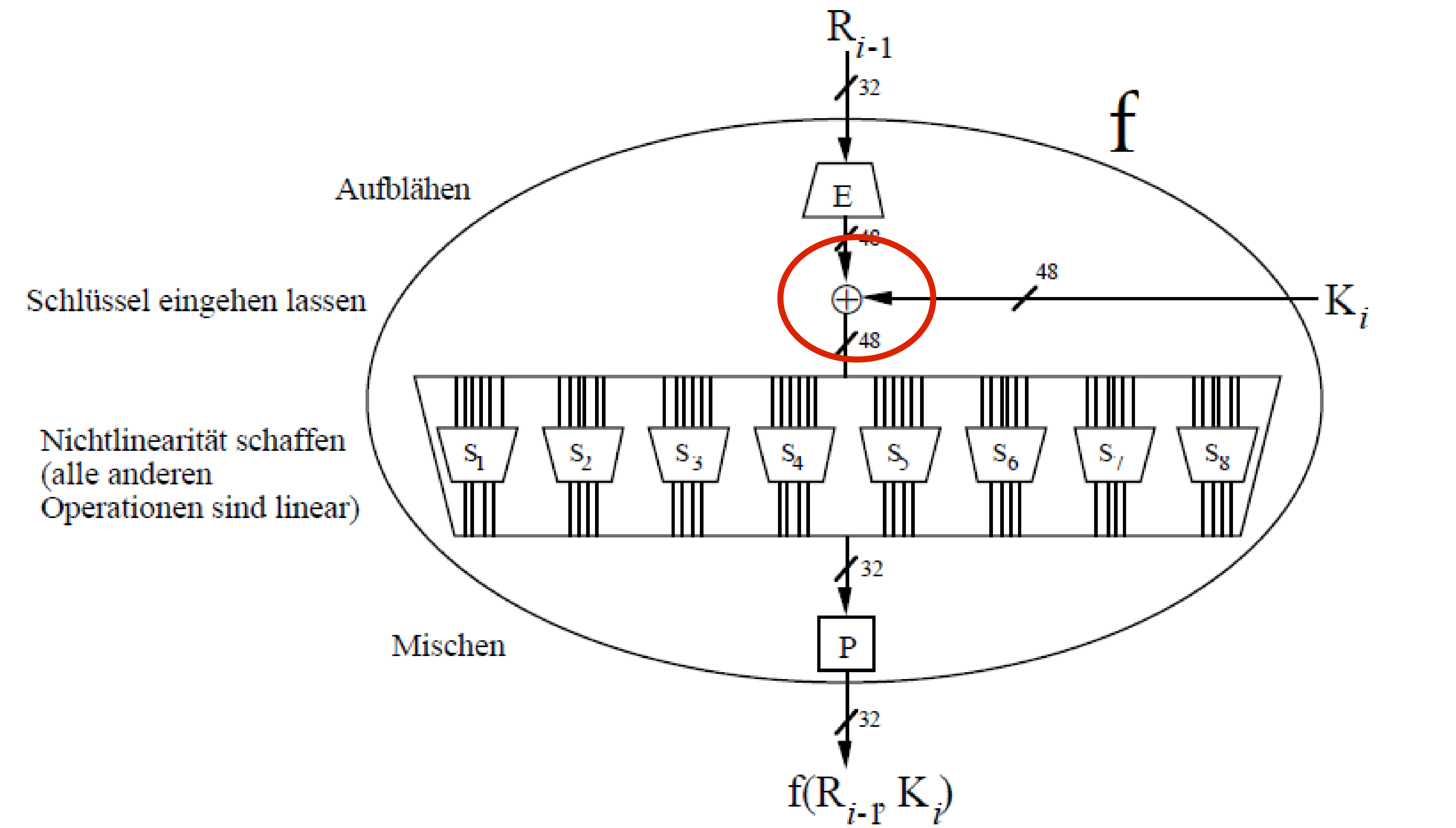


DES – die Verschlüsselungsfunktion f

- R^{i-1} wird durch eine feste Expansionsfunktion E auf 48 Bit erweitert

- E:
  - Von den 32 Bits werden 16 verdoppelt
  - Anschließend: Permutation

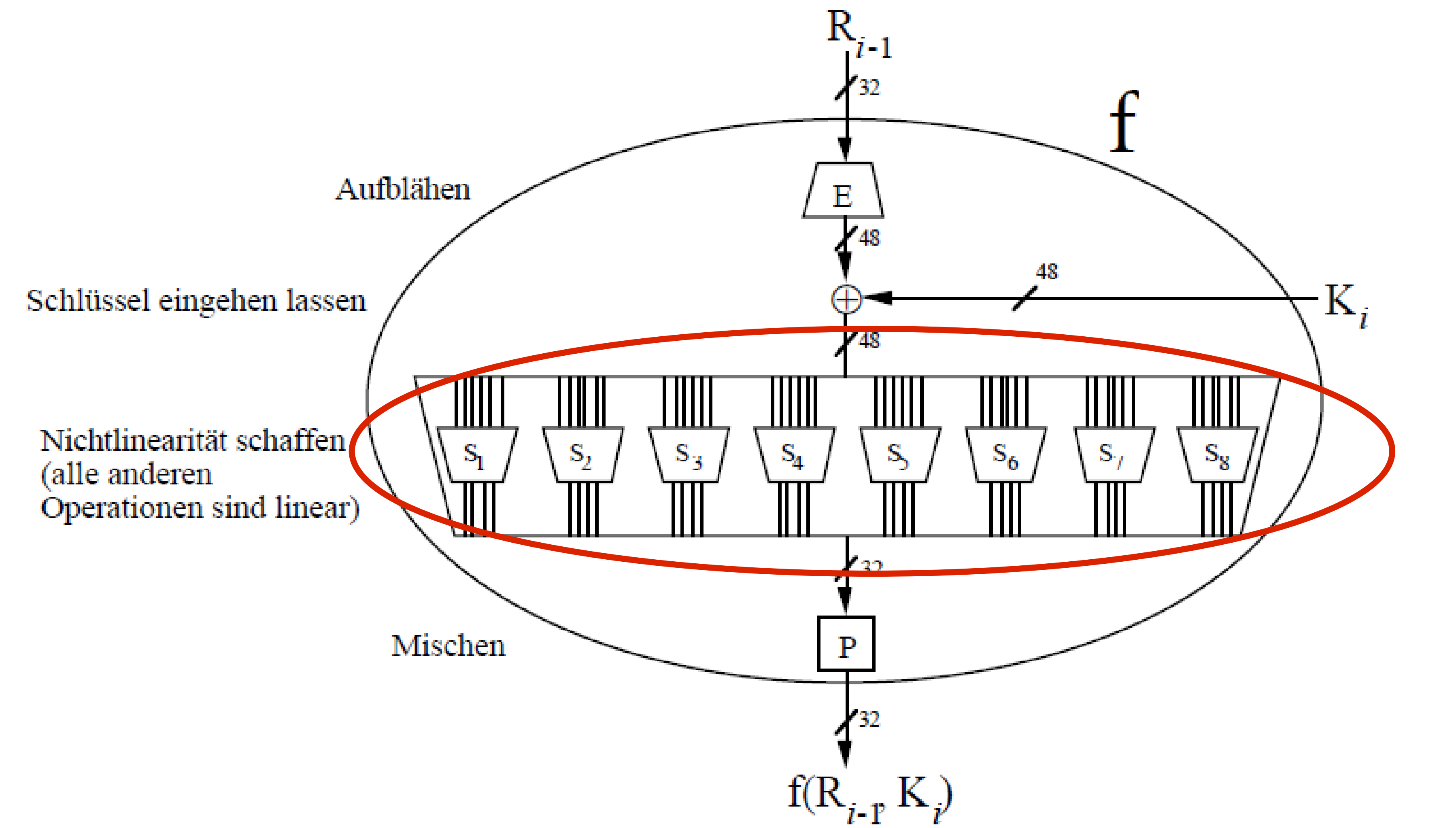
DES – die Verschlüsselungsfunktion f



DES – die Verschlüsselungsfunktion f

- $E^{(Ri-1)}$ xor K^i
- Das Ergebnis dann schon unterteilt in 8 6-Bit-Strings geschrieben

DES – die Verschlüsselungsfunktion f



DES – die Verschlüsselungsfunktion f

- 8 S-Boxen mit $S_i : \{0,1\}^6 \rightarrow \{0,1\}^4$
- In 16 * 4 Tabelle gespeichert
  - 1. und letztes Bit = Zeilennummer
  - Die 4 mittleren Bit = Spaltennummer

S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

DES – die Verschlüsselungsfunktion f

S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

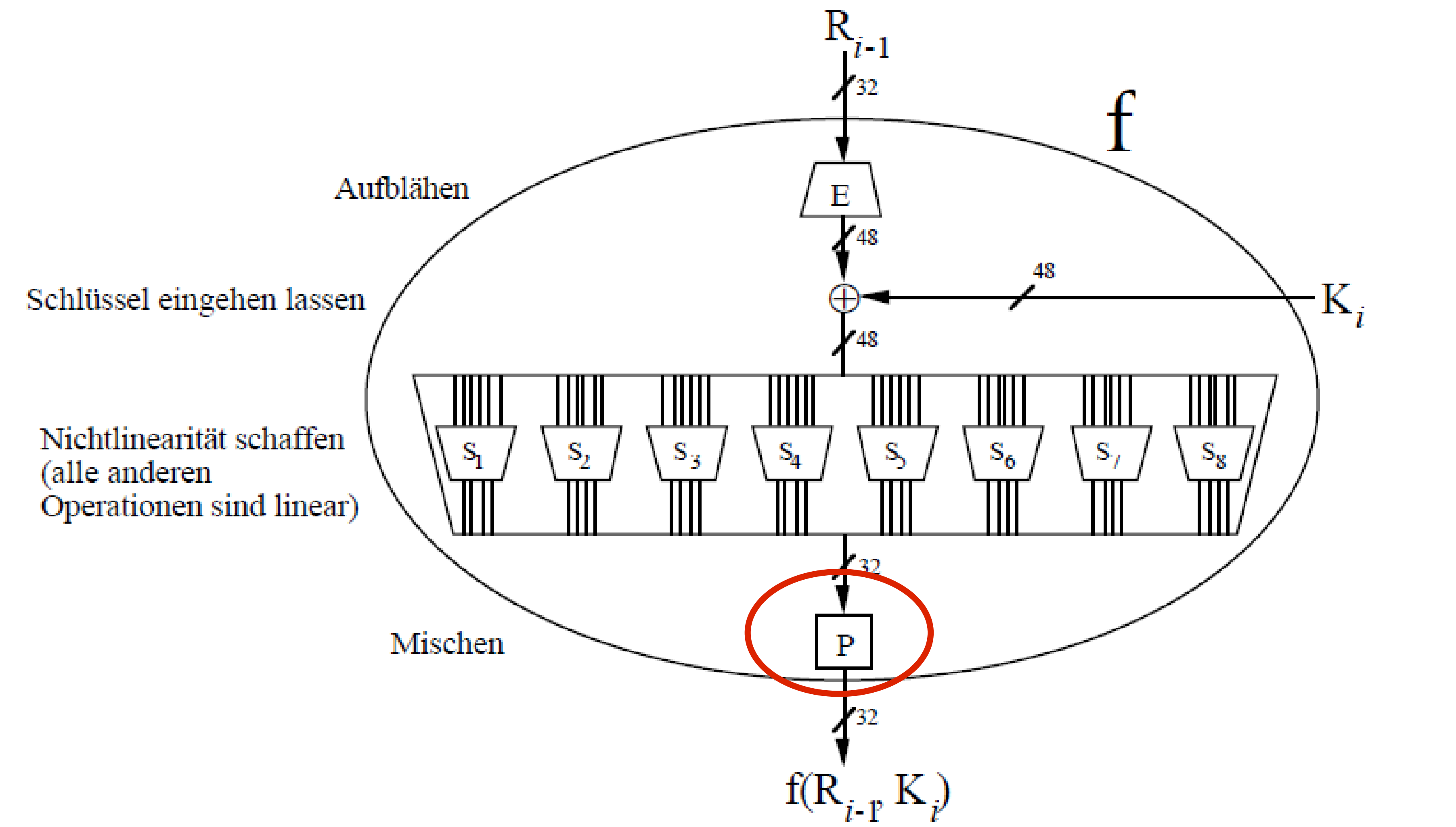
S_7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

DES – die Verschlüsselungsfunktion f



DES – die Verschlüsselungsfunktion f

- Zusammensetzen zu 32 Bit
- Permutation P

DES - Kryptoanalyse

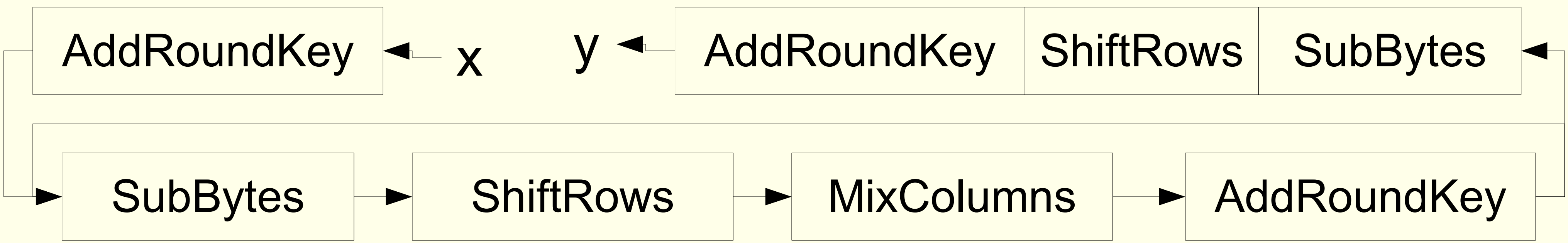
- Das Größte Sicherheitsrisiko des DES ist wohl der relativ kleine Schlüsselraum von nur 2^{56} möglichen Schlüsseln.
  - Schon 1999 wurden zum knacken eines 88 Byte langen Chiffretextes nur 22 Stunden und 15 Minuten benötigt
  - (Das Durchsuchen des gesamten Raumes ca. 82 Std.)
- S-Boxen wurden „zufällig“ zusammengestellt
  - → Gerüchte NSA hätte Hintertüren eingebaut (keine Seite nachgewiesen)

Advanced Encryption Standard

- 1997 Ausschreibung des AES
- 1998 1. AES Kandidaten Konferenz: 15 der 21 Kryptosysteme als Kandidaten zugelassen
- 1999 2. AES Kandidatenkonferenz: 5 Finalisten ausgewählt: MARS, RC6, Rijndael, Serpent, Twofish
- 2000 3. AES Kandidaten Konferenz: Rijndael = AES

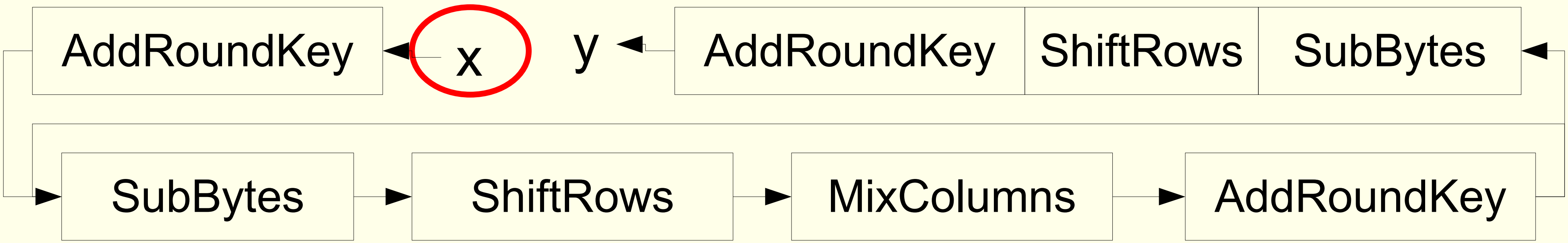
- Blocklänge: 128 Bit
- Schlüssellänge: 128, 192, oder 256 Bits
- Rundenanzahl: 10, 12, oder 14 (abhängig von Schlüssellänge)

AES - Verschlüsselung



1. Umwandeln des Klartextes in einen „State“ anschließend „AddRoundKey“
2. N-1 mal: „SubBytes“, “ShiftRows”, “MixColumns”, “AddRoundKey“
3. „SubBytes“, “ShiftRows”, “AddRoundKey“
4. Umwandeln von „State“ in Chiffretext

AES - Verschlüsselung



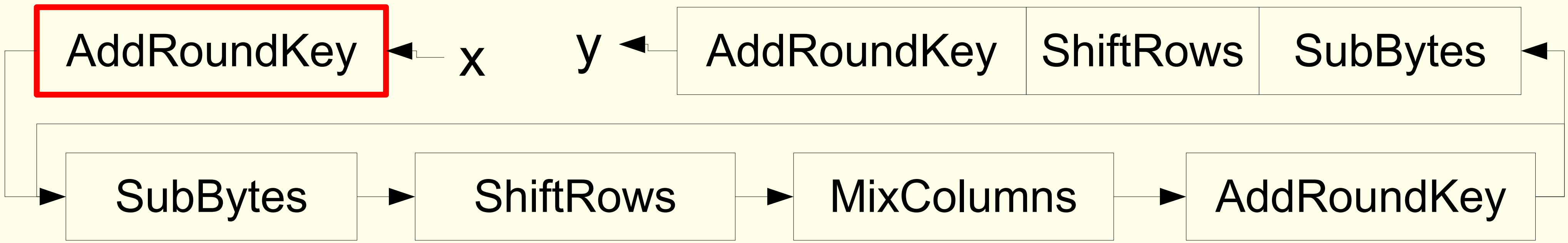
- Alle Operationen in Rijndael sind byteorientiert
  - Unterteilung des Klartextes in 16 1-Byte Blöcke

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

$\longleftarrow$

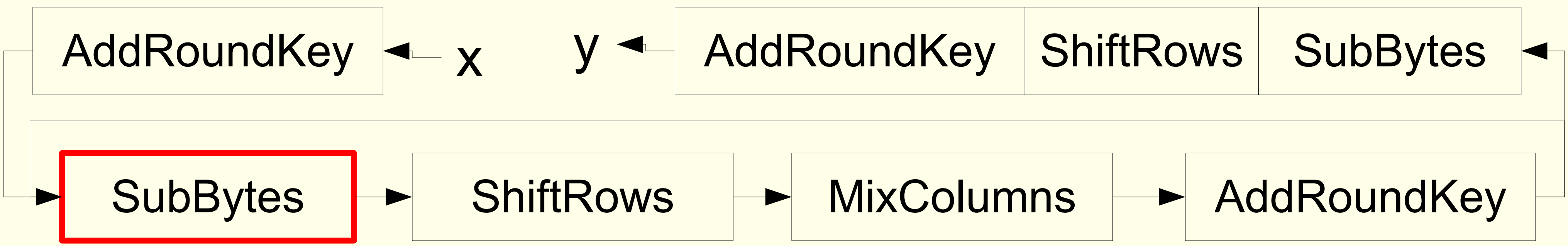
X_0	X_4	X_8	X_{12}
X_1	X_5	X_9	X_{13}
X_2	X_6	X_{10}	X_{14}
X_3	X_7	X_{11}	x_{15}

AES - Verschlüsselung



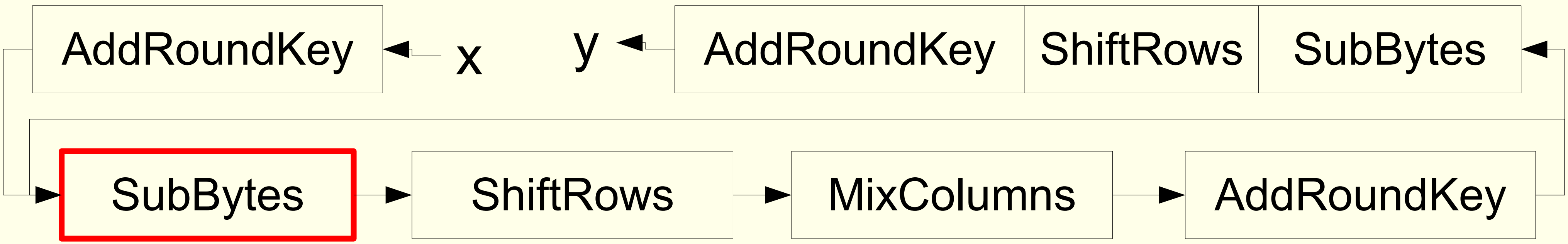
- Rundenschlüssel hat gleiche Anzahl von Byes wie „State“
- Byteweises Xor-Verknüpfen der beiden

AES - Verschlüsselung



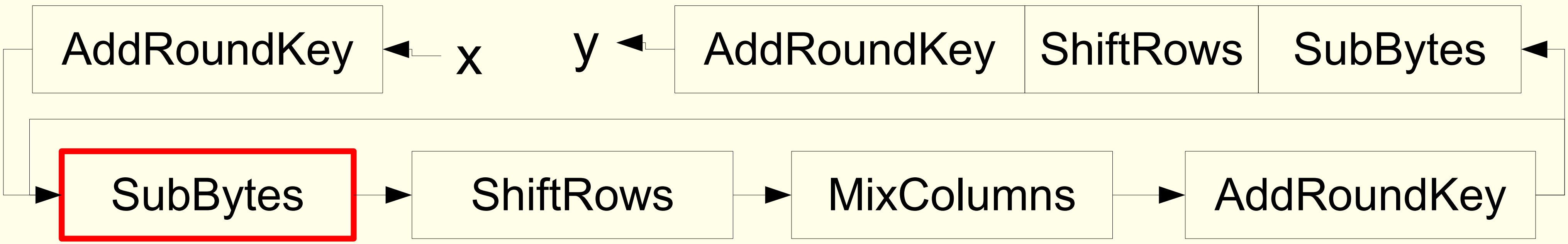
- S-Box $\pi_S : \{0,1\}^8 \rightarrow \{0,1\}^8$
- Meist als 16 x 16 Array gespeichert
- Diese S-Box ist im Gegensatz zu denen vom DES algebraisch definiert
- Basiert auf Operationen des endlichen Körpers GF(2^8)

AES - Verschlüsselung



- Z_2 ist der Restklassenring modulo 2
  - Da 2 eine Primzahl ist, ist Z_2 auch ein Körper
- π_S operiert auf dem Polynomring in x über Z_2: $Z_2[x]$
  - Jedoch nur auf Polynomen vom Grad 7 oder kleiner
  - → Bilden von Restklassen modulo eines irreduziblen Polynoms 8. Grades

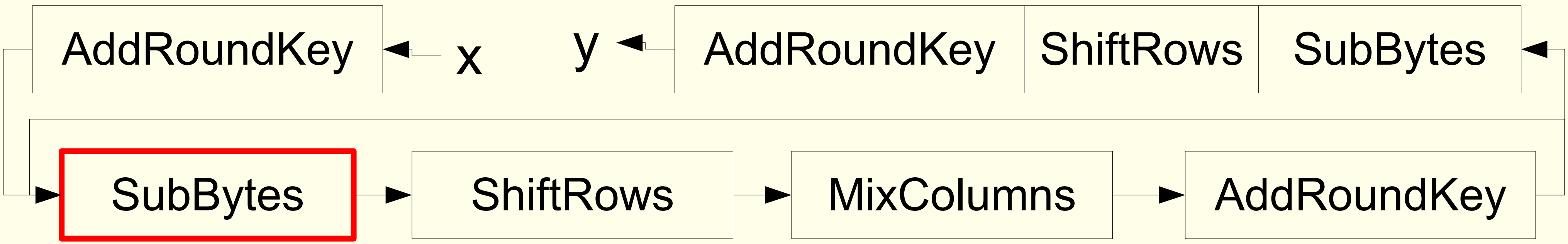
AES - Verschlüsselung



- Welches irreduzible Polynom 8. Grades ist egal
  - AES nutzt: $x^8 + x^4 + x^3 + x + 1$

$$GF(2^8) = Z_2[x]/(x^8 + x^4 + x^3 + x + 1)$$

AES - Verschlüsselung



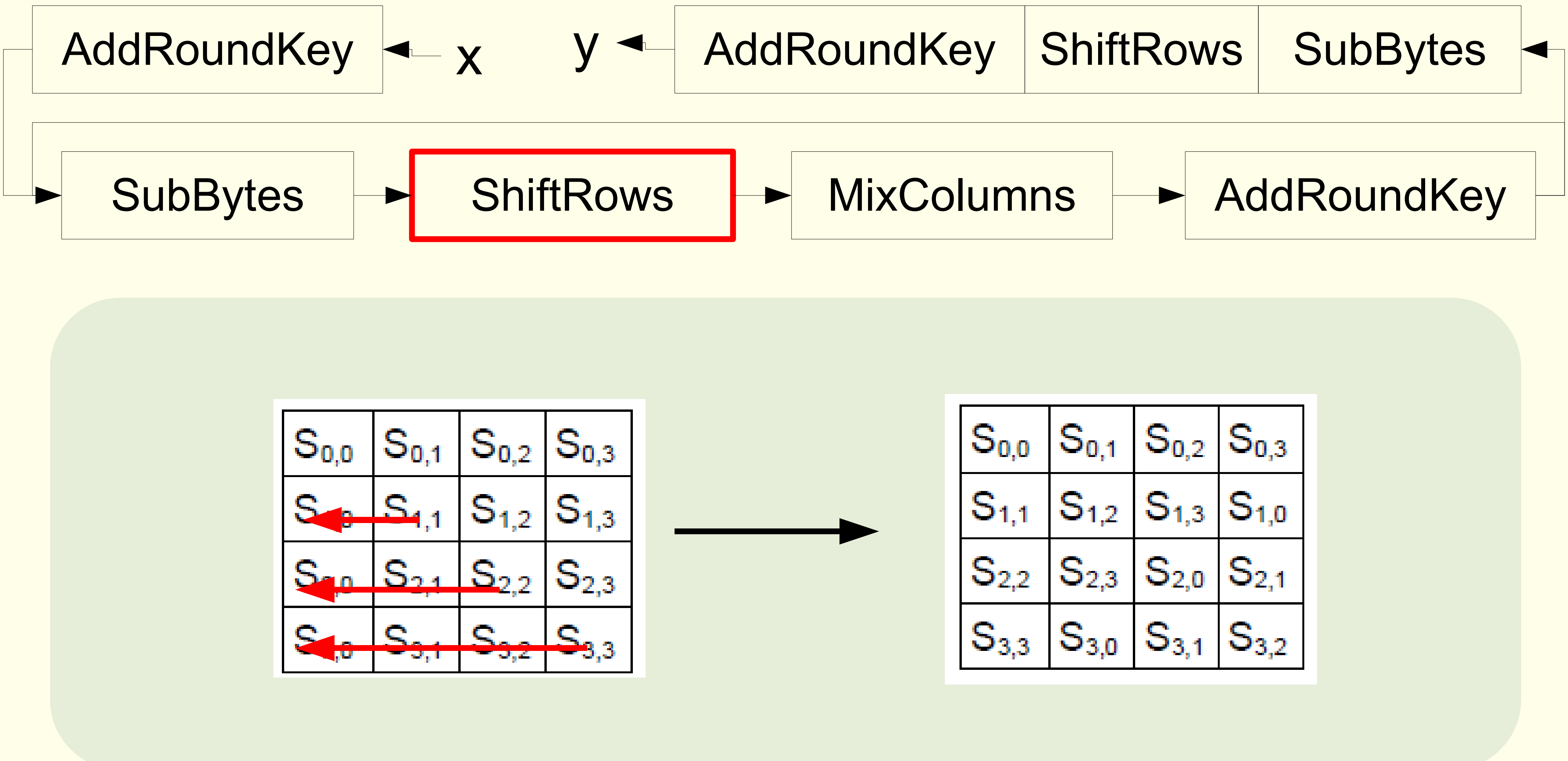
1. Byte in Polynom umwandeln
2. Falls Polynom != 0, berechne das Multiplikativ Inverse
3. Polynom in Byte umwandeln
4. Multipliziere mit Matrix M
5. Xor mit a= 11000110

$$M=\begin{bmatrix}1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\\1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\end{bmatrix}$$

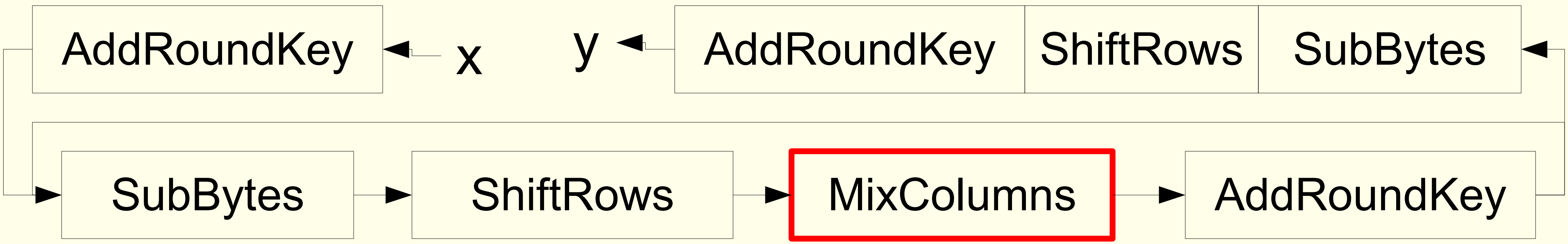
AES - Verschlüsselung

		y															
		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
x	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

AES - Verschlüsselung



AES - Verschlüsselung



- Die Spalten von „State“ werden als Polynome aufgefasst
- Multiplikation mit Matrix A (Multiplikation in $GF(2^8)$)
- A =

$$\begin{pmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{pmatrix}$$

AES – Schlüsselerzeugung (128 Bit Schlüssel)

- Die Schlüsselerzeugung ist wortbasiert (1 Wort = 4 Bytes)
- Ein Rundenschlüssel besteht aus 4 Wörtern
- 128 Bit-Schlüssel → 10 Runden → 11 Rundenschlüssel
- Festes Array Rcon bestehend aus 10 Wörtern
- Teilfunktionen:
  - RotWord: rotiert die Bytes eines Wortes um eine Stelle nach links
  - SubWord: nutzt für jedes Byte die S-Box aus SubBytes
  - Xor

AES – Schlüsselerzeugung (128 Bit Schlüssel)

RCon (in Hexadezimaldarstellung):

01 00 00 00

02 00 00 00

04 00 00 00

08 00 00 00

10 00 00 00

20 00 00 00

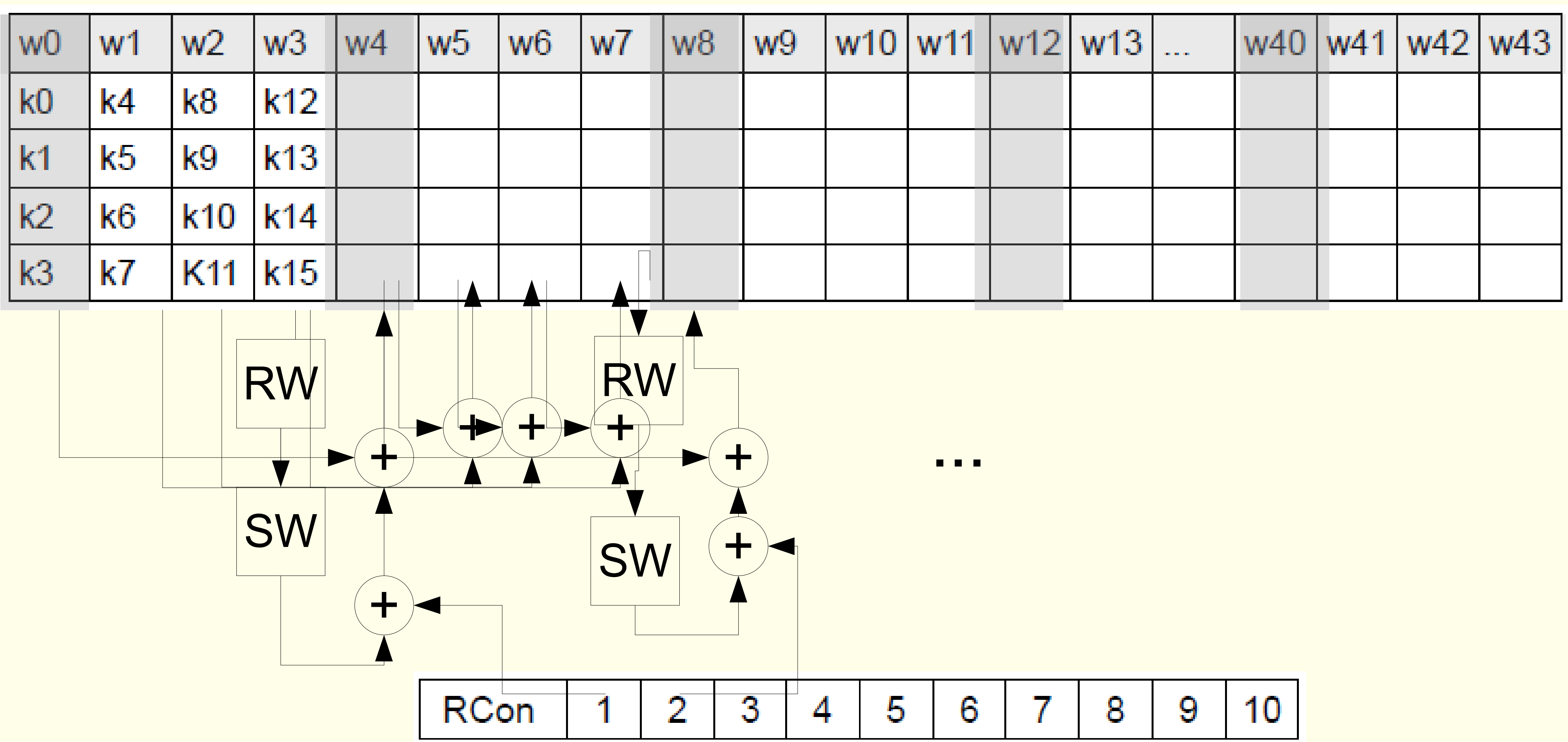
40 00 00 00

80 00 00 00

1B 00 00 00

36 00 00 00

AES – Schlüsselerzeugung (128 Bit Schlüssel)



AES - Entschlüsselung

- Die Reihenfolge der Operationen ist umgekehrt
- Dabei werden anstelle von ShiftRows, SubBytes und MixColumns ihre inversen Funktionen genutzt
- Außerdem werden die Rundenschlüssel in umgekehrter Reihenfolge verwendet

AES - Kryptoanalyse

- Die Entwickler des Algorithmus selber, wiesen schon nach, dass Rijndael gegen lineare und differentielle Kryptoanalyse resistent
- Bis heute keine effektiven Angriffe auf Rijndael bekannt
- Die effektivsten Varianten lehnen sich an eine Chiffre mit einer reduzierten Rundenzahl an
- Doch selbst diese sind kaum besser als einfache Suche

Referenzen

- Douglas R. Stinson: Cryptography: Theory and Practice. 3nd Edition, Chapman & Hall/CRC 2006
- Andreas Pfitzmann: Sicherheit in Rechnernetzen: Mehrseitige Sicherheit in verteilten und durch verteilte Systeme (http://www.inf.tu-dresden.de/index.php?node_id=510&ln=de)
- Albrecht Beutelspacher, Heike B. Neumann, Thomas Schwarzpaul: Kryptografie in Theorie und Praxis: mathematische Grundlagen für elektronisches Geld, Internetsicherheit und Mobilfunk Vieweg+Teubner Verlag, 2005

Referenzen

- Howard M.Heys: A tutorial on linear and differential cryptanalysis (Faculty of Engineering and Applied Science - Memorial University of Newfoundland)
-