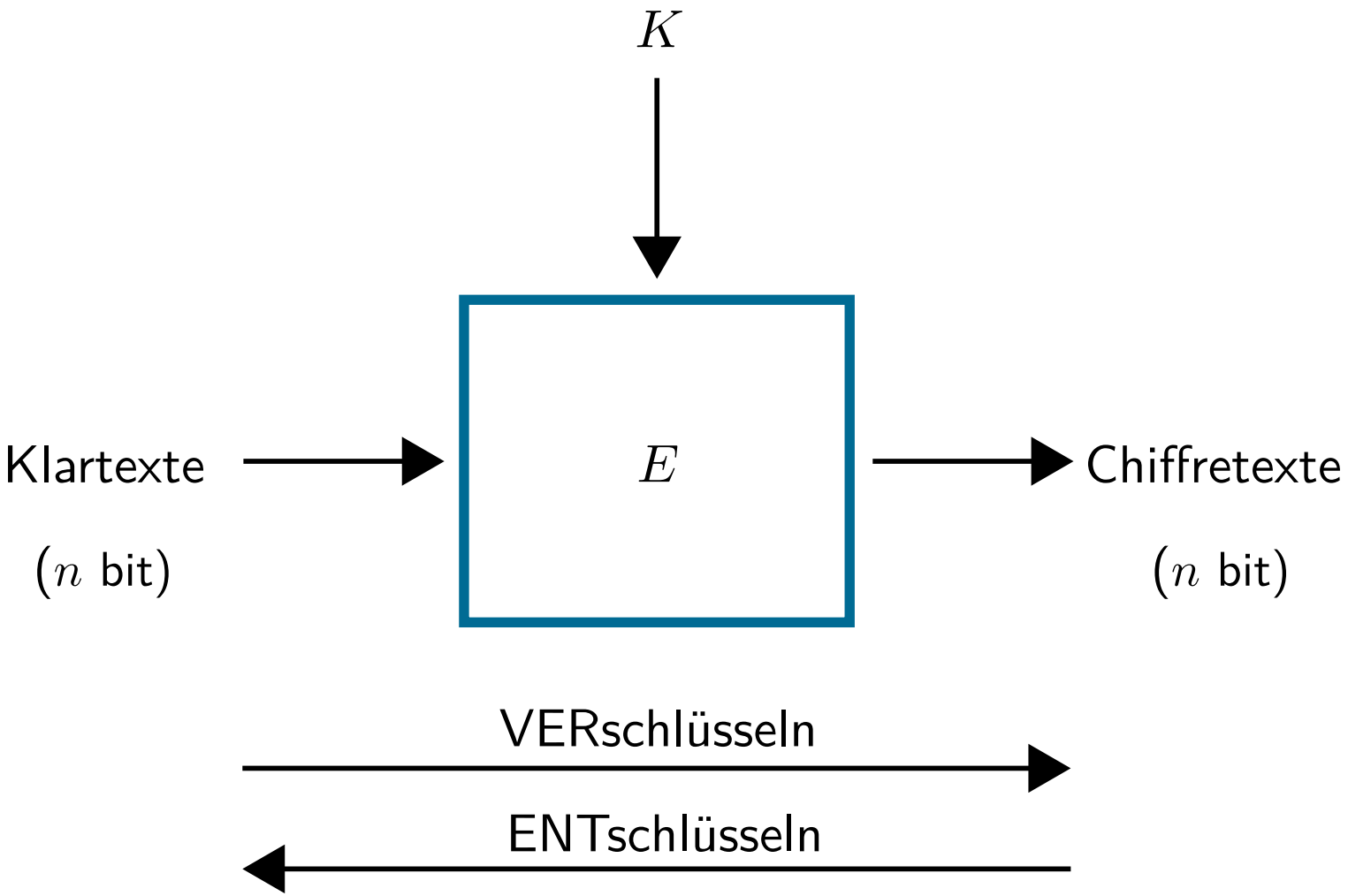
4: Blockchiffren



4.1: Abstrakte Blockchiffren

- Familie von Paaren (E, D) effizient berechenbarer Funktionen

$$E_{\mathbf{K}}, D_{\mathbf{K}} : \{0,1\}^n \to \{0,1\}^n.$$

  (Es sind E die Ver- und D die Entschlüsselungsoperation und $\mathbf{K}$ der Schlüssel.)
- Für jeden Schlüssel $\mathbf{K} \in \{\mathbf{0},\mathbf{1}\}^{\mathbf{k}}$ und jeden Klartext $x \in \{0,1\}^n$ muss gelten:

$$D_{\mathbf{K}}(E_{\mathbf{K}}(x)) = x.$$

Ist das eine Chiffre?

- Klartextmenge = Chiffretextmenge = $\{0,1\}^n$
- Schlüsselmenge = $\{\mathbf{0},\mathbf{1}\}^{\mathbf{k}}$
- Verschlüsselungsoperation E
- Entschlüsselungsoperation D
- Schlüsselerzeugung typischerweise trivial

- Seien die Werte ("Sicherheitsparameter") n und $\mathbf{k}$ gegeben. Wieviele verschiedene abstrakte Blockchiffren gibt es für n und $\mathbf{k}$?

Es gibt $(2^n!)^{2^k}$ verschiedene Blockchiffren.

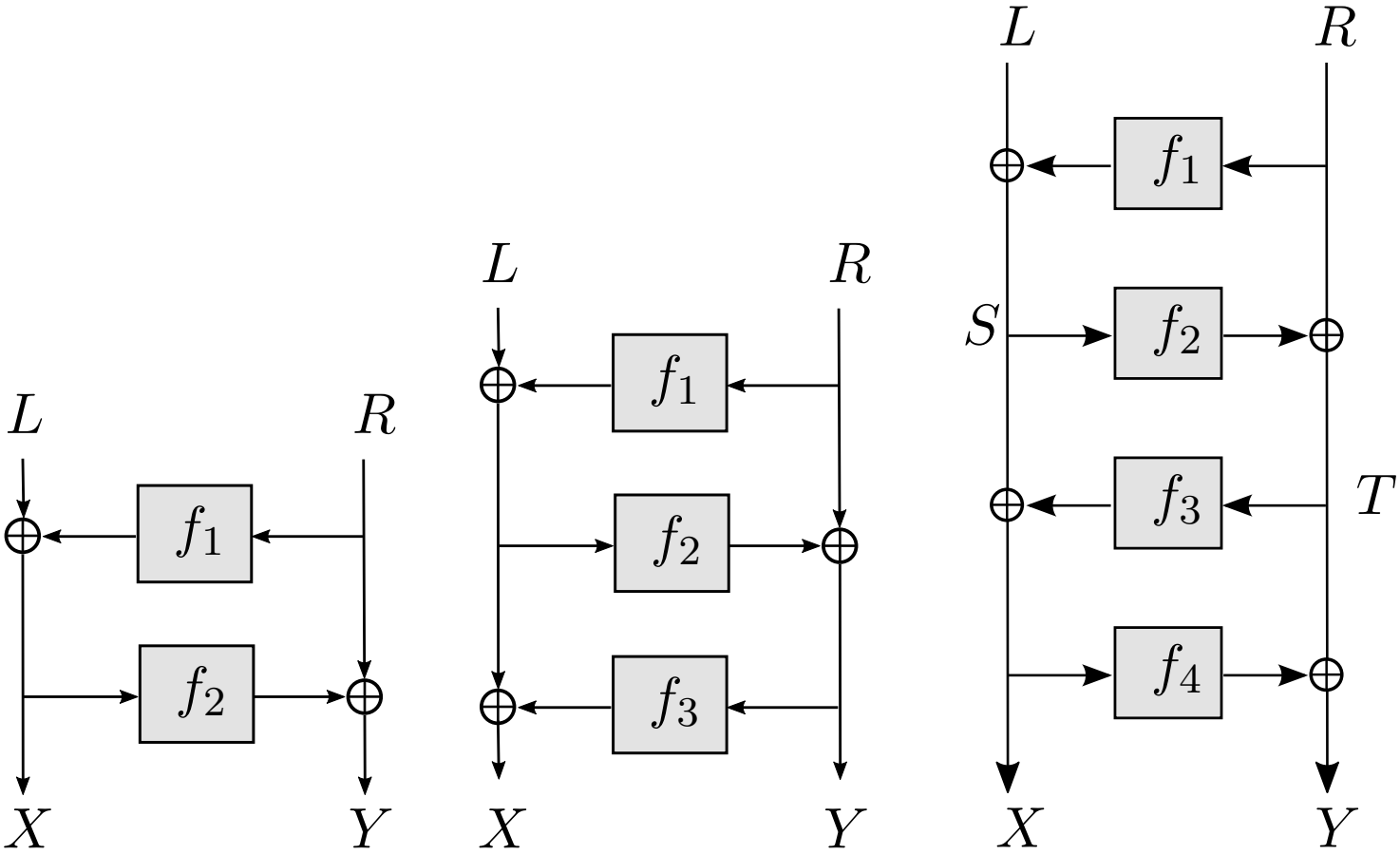
Sicherheit: Chosen Plaintext Angr. auf einen PZPG (Pseudozufalls-Permutationsgenerator)

Ein *Chosen-Plaintext-Angreifer* auf einen PZPG ist ein Algorithmus mit "Orakel-Zugriff" auf eine Permutation $E : \{0,1\}^n \to \{0,1\}^n$. Zwei mögliche Fälle:

- "Fall 0": E ist eine Zufallspermutation
- "Fall 1": E ist eine Pseudozufallspermutation

Ein PZBG ist *sicher gegen Chosen-Plaintext-Angreifer*, wenn es keinen effizienten Angreifer gibt, der mit signifikantem Vorteil zwischen den Fällen 0 und 1 unterscheiden kann.

Luby-Rackoff/Feistel-Chiffren



- $P_2,\ P_3,\ P_4$

Die Blockchiffre P_2

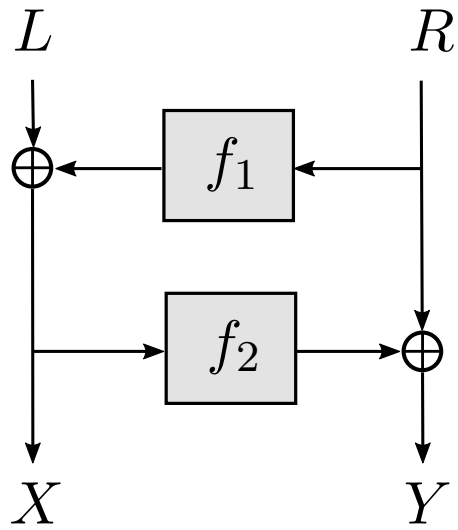
Seien $f_1, f_2 : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ Zufallsfunktionen. Wir betrachten:

$\mathbf{P_2}(\mathbf{L}, \mathbf{R})$:

1: $X \leftarrow L \oplus f_1(R)$

2: $Y \leftarrow R \oplus f_2(X)$

3: **return** (X, Y)



Beachte: $P_i : \{0,1\}^n \to \{0,1\}^n$, aber $f_j : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$

Frage 1: Ist P_2 tatsächlich eine Permutation?

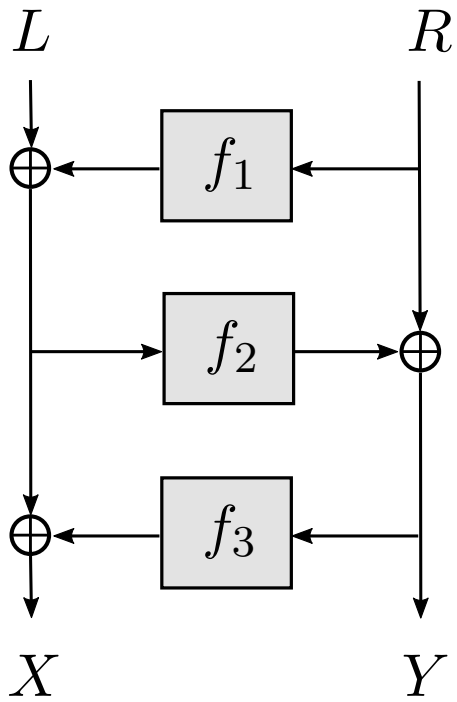
Frage 2: Ist P_2 sicher gegen Chosen-Plaintext-Angreifer? ($\to$ Tafel)

Die Blockchiffre P_3

Seien $f_1, f_2, f_3 : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ Zufallsfunktionen. Wir betrachten:

$\mathbf{P_3}(\mathbf{L}, \mathbf{R})$:

1: $S \leftarrow L \oplus f_1(R)$
2: $Y \leftarrow R \oplus f_2(S)$
3: $X \leftarrow S \oplus f_3(Y)$
4: **return** (X, Y)



Die Blockchiffre P_3

Satz 13 (Luby und Rackoff)

P_3 ist eine Permutation und sicher gegen Chosen-Plaintext-Angreifer.

(Ineffizienter) Chosen-Plaintext Angriff:

1. Wähle Klartexte (L_i, R_i) mit $R_i \neq R_j$ bis $i < j$ mit $Y_i \oplus Y_j = R_i \oplus R_j$.

   Statistisch zu erwarten: $q \approx 2^{n/4}$ Klartexte ($\rightarrow$ Tafel).

   Für den P_3 gilt $S_i = S_j$ oder ($S_i \neq S_j$ und $f_3(S_i) = f_3(S_j)$) ($\rightarrow$ Tafel).
2. Wähle zwei weitere Klartexte $(L_i \oplus \delta, R_i), (L_j \oplus \delta, R_j)$.

   Falls P_3 und $S_i = S_j$: ($\rightarrow$ Tafel).

Zweiseitige Angreifer (Chosen Ciphertext)

Ein *zweiseitiger Angreifer* (oder *Chosen Ciphertext Angreifer*) auf einen PZPG ist ein Algorithmus mit "Orakel-Zugriff" auf eine Permutation $E : \{0,1\}^n \to \{0,1\}^n$ und ihre Umkehrung E^{-1}.

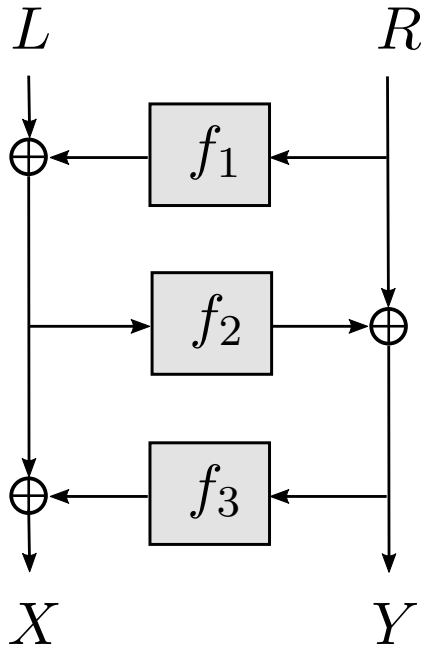
Es liegt einer der beiden folgenden Fälle vor:

- "Fall 0": E ist eine Zufallspermutation
- "Fall 1": E ist eine Pseudozufallspermutation

Ein PZPG ist *sicher gegen zweiseitige Angreifer*, wenn es keinen effizienten Angreifer gibt, der mit signifikantem Vorteil zwischen den Fällen 0 und 1 unterscheiden kann.

Zweiseitige Sicherheit von P_3

Ist P_3 sicher gegen zweiseitige Angreifer?
($\rightarrow$ Tafel)

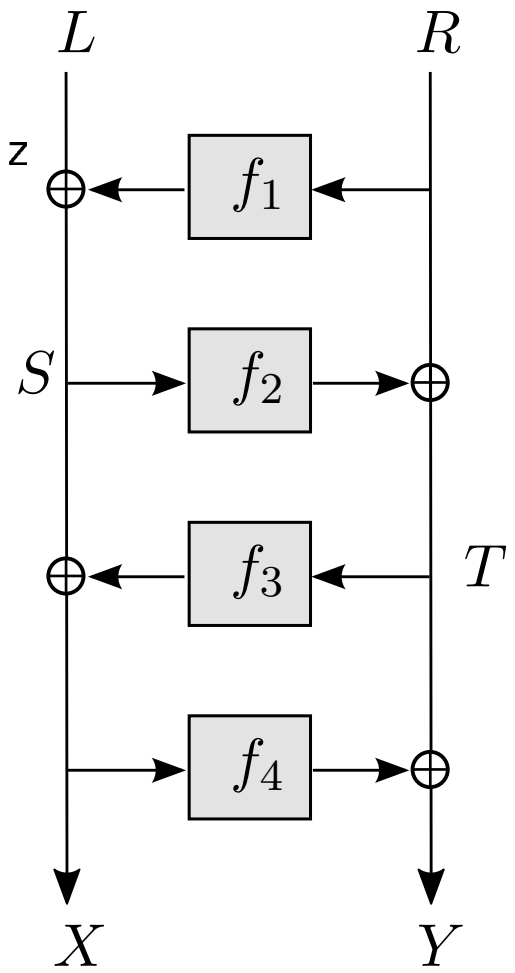


Die Blockchiffre P_4

Seien $f_1, f_2, f_3, f_4 : \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ Zufallsfunktionen. Wir betrachten:

$\mathbf{P_4(L,R)}$:

1: $S \leftarrow L \oplus f_1(R)$
2: $T \leftarrow R \oplus f_2(S)$
3: $X \leftarrow S \oplus f_3(T)$
4: $Y \leftarrow T \oplus f_4(X)$
5: **return** (X, Y)



Satz 14 (Luby und Rackoff)

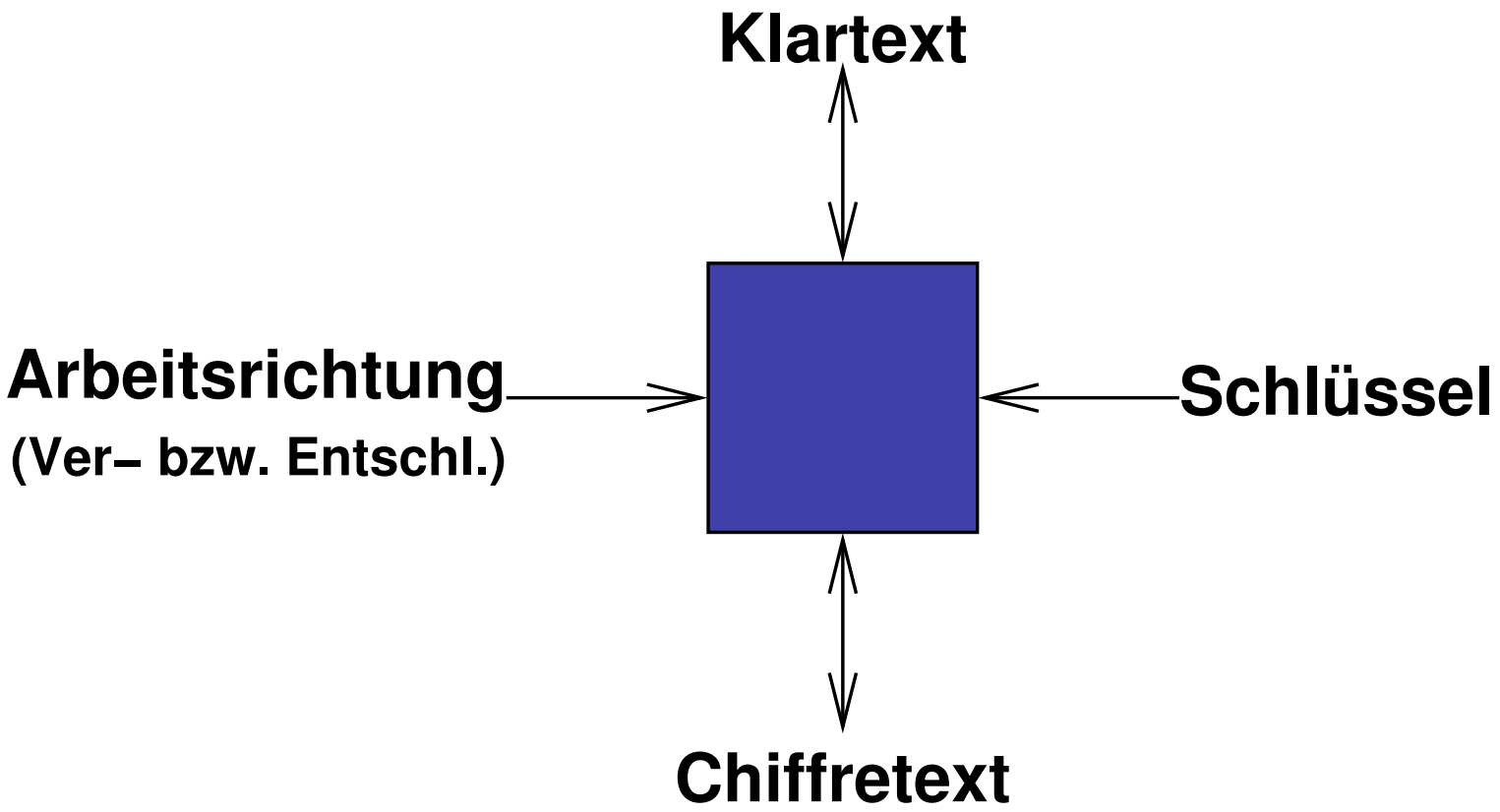
P_4 ist sicher gegen zweiseitige Angreifer.

Zwischenbemerkungen

Sie sollten

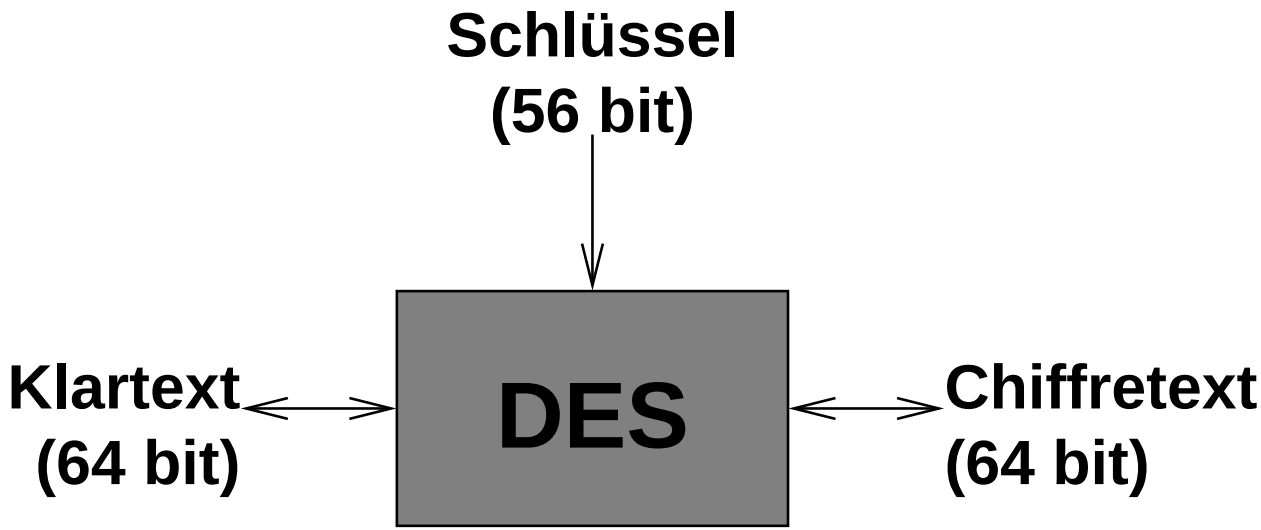
- wissen, was eine Blockchiffre ist,
- wissen, was eine Luby-Rackoff-Chiffre (Feistelchiffre) ist,
- Sicherheitskriterien für Blockchiffren kennen,
- und die Sicherheit bzw. Unsicherheit der Luby-Rackoff-Chiffren P_2, P_3, P_4 und verwandter Konstuktionen für abstakte Blockchiffren abschätzen können (mit Begründung).

Von abstrakten zu konkreten Blockchiffren



Wichtige Parameter: Blockgröße n, Schlüssellänge $\mathbf{k}$

4.2: Der Data Encryption Standard (DES)



Geschichte des DES

1973-77 Zwei Ausschreibungen, ein geeigneter Kandidat ("Lucipher") nach Überarbeitung als DES ("Data Encryption Standard") standardisiert:

64-bit Blockchiffre mit 56-bit Schlüsseln.

Ab 1977 Kritik an Schlüssellänge.
Trotzdem große Akzeptanz und riesige Verbreitung.

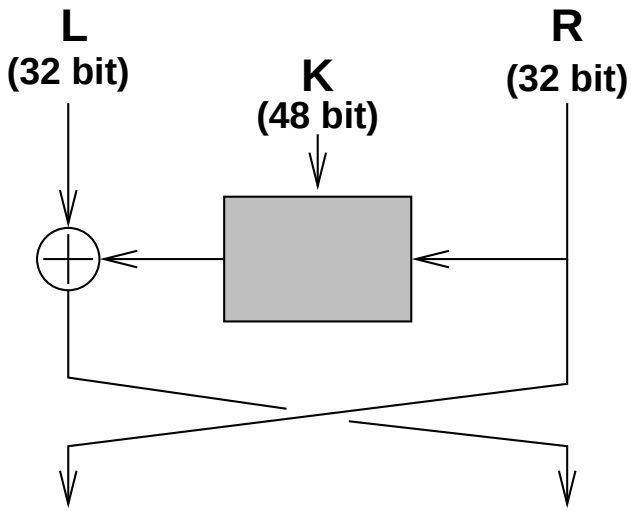
Ab 1990 Differentielle und lineare Kryptanalyse.

1997 DES-Challenge (1000e von Rechnern, 4 Mon.).

Struktur des DES (Feistel-Netzwerk)

- Rundenfunktion:

$$f : \{0,1\}^{48} \times \{0,1\}^{32} \to \{0,1\}^{32}$$



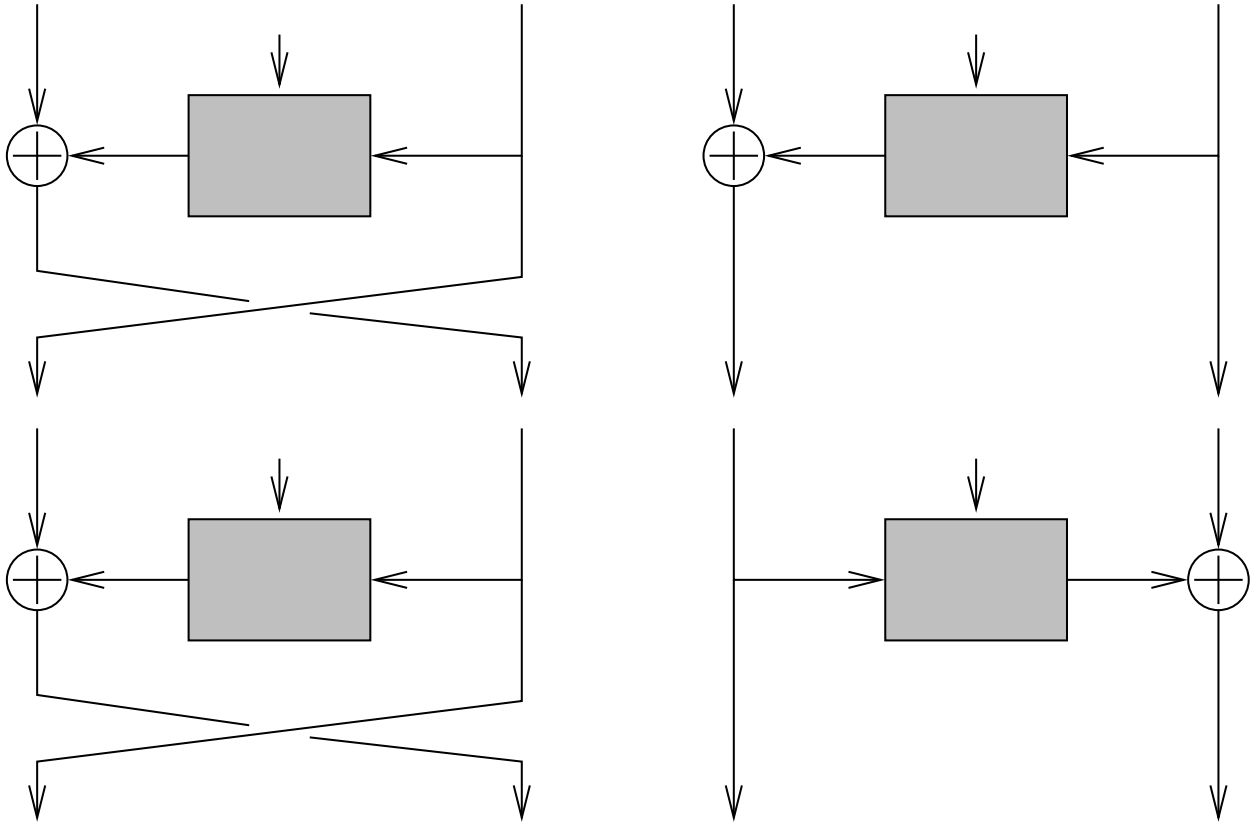
- 16 Runden
- 16 Rundenschlüssel

$$\mathbf{K}[1], \ldots, \mathbf{K}[16] \in \{\mathbf{0},\mathbf{1}\}^{\mathbf{48}},$$

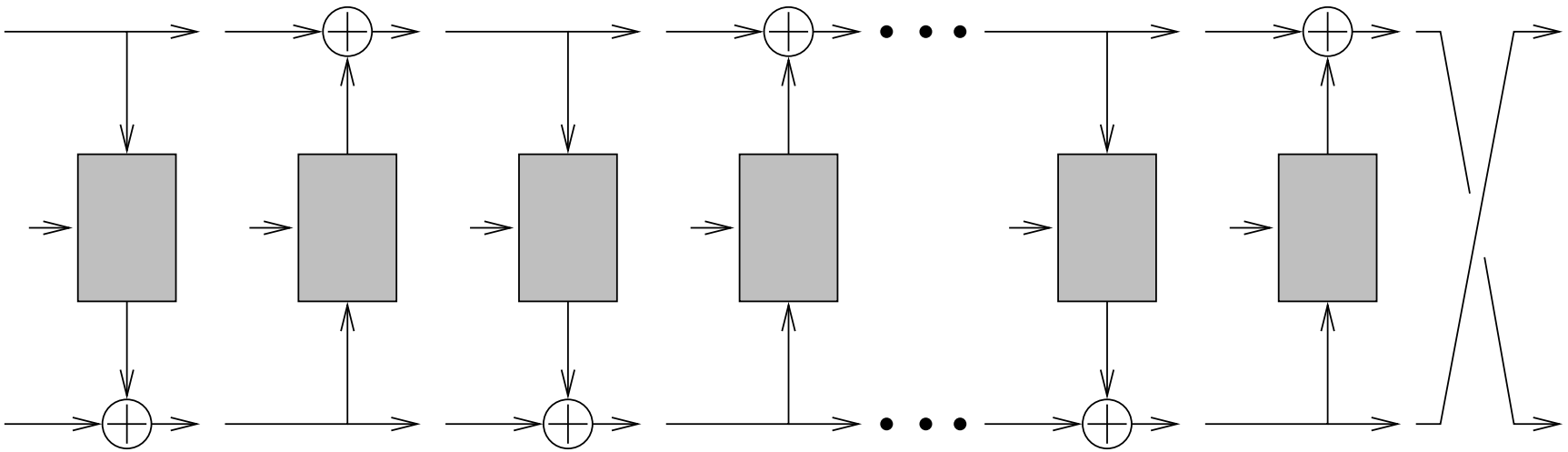
abgeleitet aus einem 56-bit Chiffrierschlüssel.

Diese "Feistel-Chiffre" ist die Verallgemeinerung der abstrakten Blockchiffren P_2, P_3 und P_4.

Zwei verschiedene Darstellungweisen



DES: Insgesamt 16 Runden



Zusätzlich zur Rundenfunktion

- Anwendung einer schlüsselunabhängigen "Initial Permutation" ("wire crossing")

$$\mathsf{IP} : \{1, \ldots, 64\} \to \{1, \ldots, 64\}$$

  am Anfang. Anwendung von IP^{-1} am Ende.

$$DES_{\mathbf{K}}(M) := \mathsf{IP}^{-1}(f_{\mathbf{K}[16]}(\cdots(f_{\mathbf{K}[1]}(\mathsf{IP}(M))))).$$

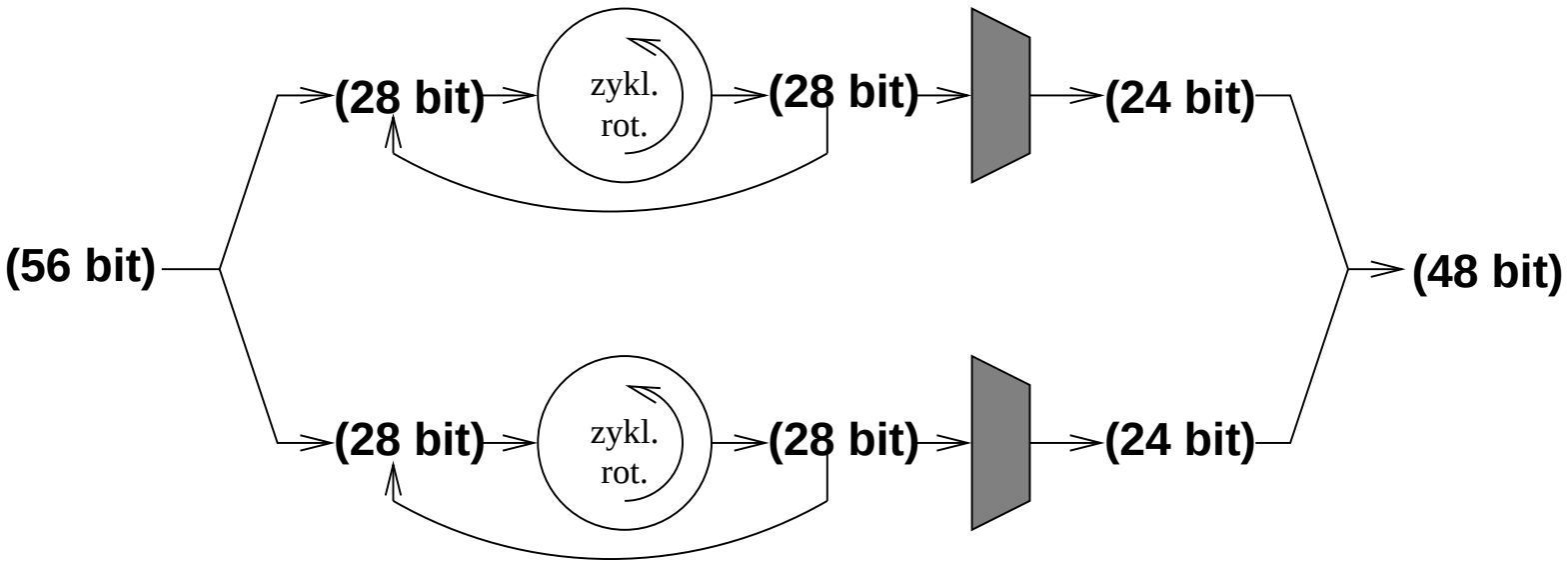
- In Hardware ist das praktisch "kostenlos", in Software typischerweise etliche Rechenschritte bzw. Takte.
- Der Sinn von IP und IP^{-1} ist unklar. Für die Sicherheit des DES sind beide irrelevant. (**Warum?**)
- Wir können $\mathsf{IP}/\mathsf{IP}^{-1}$ ignorieren.

Wie Entschlüsselt man?

($\rightarrow$ Tafel)

Der DES Key-Schedule

Der Key-Schedule nimmt 56 Schlüsselbits als Eingabe und produziert 16 Rundenschlüssel zu jeweils 48 bit.



Der DES Key-Schedule (2)

- **null**, **eins** $\in \{0,1\}^{28}$ bezeichnen die Konstanten $\mathbf{0}\ldots\mathbf{000}$ und $\mathbf{1}\ldots\mathbf{111}$.
- Ist eine Hälfte von $\mathbf{K}$ entweder gleich **null** oder gleich **eins**, dann verändert sie sich im Verlauf des Key-Schedules nicht.
- Sinde beide Hälften gleich **null** oder gleich **eins**, d.h. $\mathbf{K} \in \{(\textbf{null},\textbf{null}),(\textbf{null},\textbf{eins}),(\textbf{eins},\textbf{null}),(\textbf{eins},\textbf{eins})\}$, dann gilt:

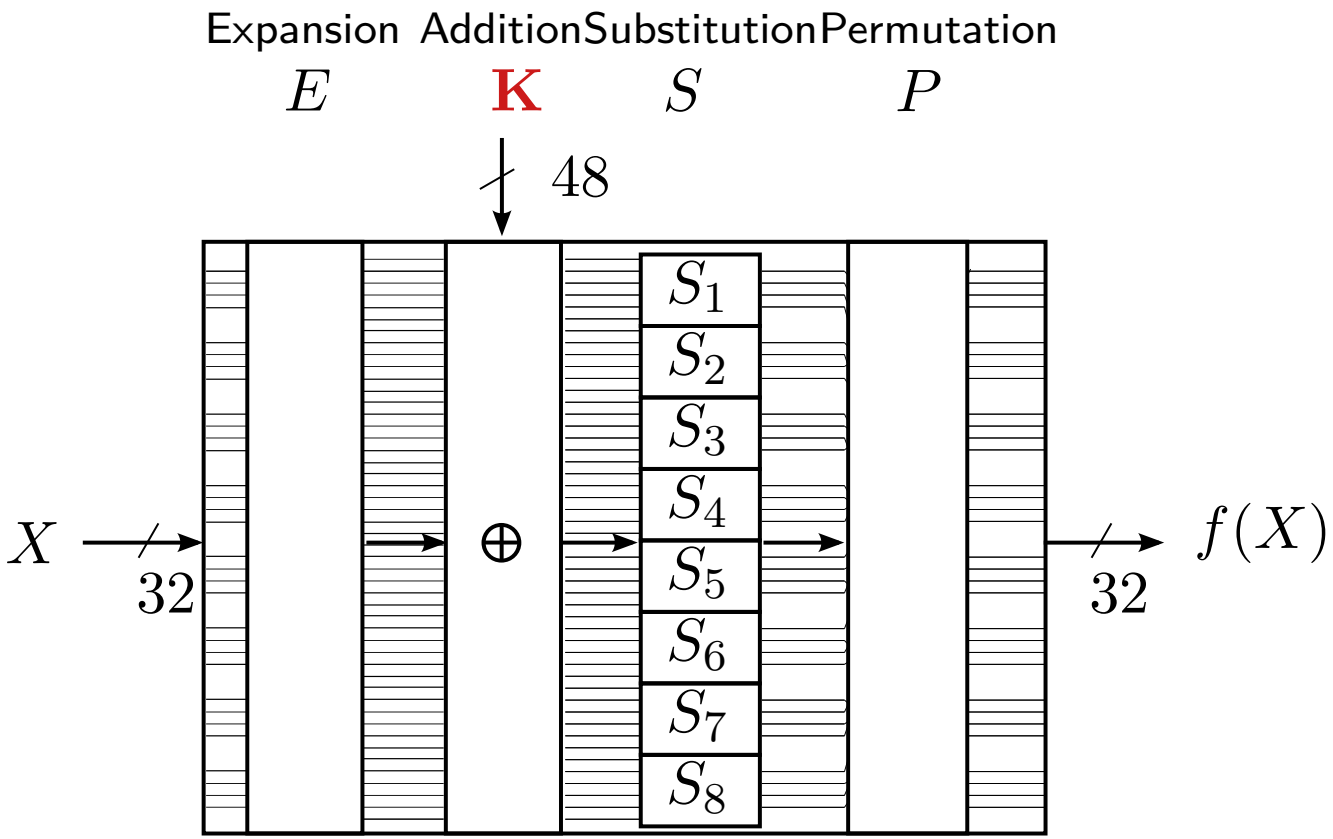
$$\mathbf{K}[1] = \mathbf{K}[2] = \cdots = \mathbf{K}[16].$$

Der DES Key-Schedule (3)

- Für diese vier Schlüssel $\mathbf{K}$ gilt: $E_{\mathbf{K}} = D_{\mathbf{K}}$.
- Derartige Schlüssel bezeichnet man als schwach.
- Man kennt keine weiteren schwachen Schlüssel.
- Außerdem kennt man 6 Paare semi-schwacher Schlüssel. Dies sind Paare $(\mathbf{K}, \mathbf{L})$ mit $E_{\mathbf{K}} = D_{\mathbf{L}}$.

Die f-Funktion des DES

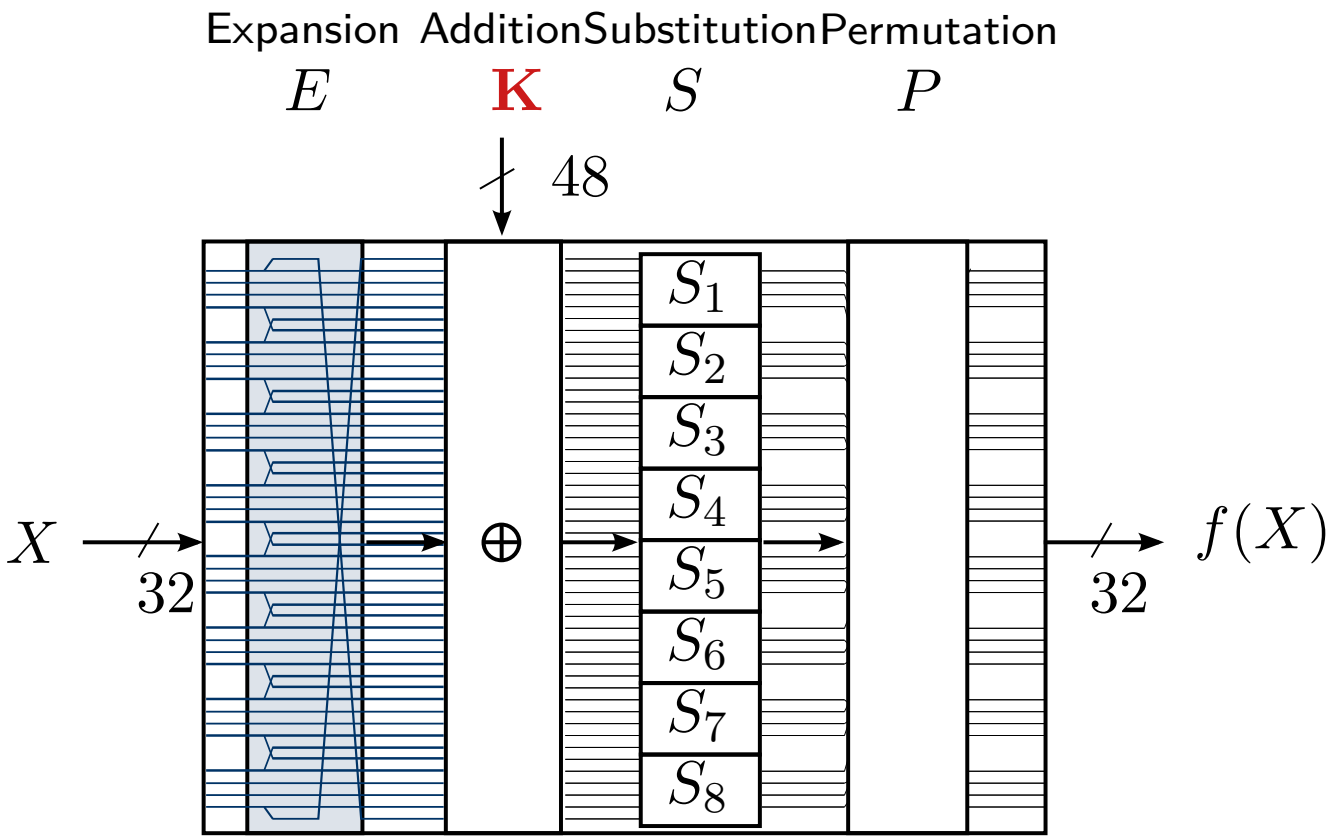
Die f-Funktion im Detail



$$f_{\mathbf{K}[i]}(X) := P(S(E(X) \oplus \mathbf{K}[i])).$$

Die Expansionsfunktion (E)

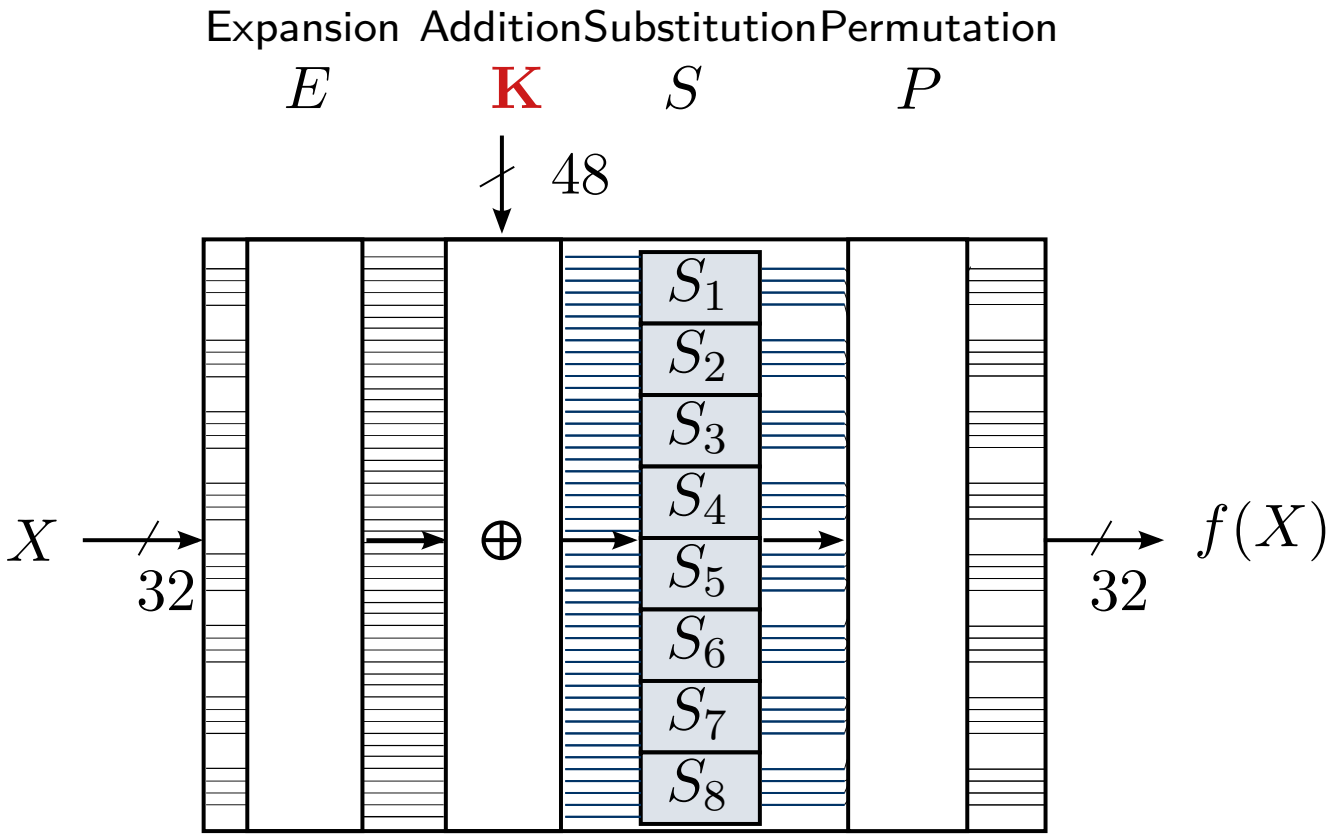
Die Expansionsfunktion $E : \{0,1\}^{32} \to \{0,1\}^{48}$ expandiert 32 zu 48 Bits.



Die Substitution (S)

Die acht Substitutionsboxen (S-Box $S_1, \ldots, S_8$) ersetzen jeweils sechs Eingabe- durch vier Ausgabebits:

$$S(X) : (\{0,1\}^6)^8 \to (\{0,1\}^4)^8$$



Die Substitution: S-Box 1 (S_1)

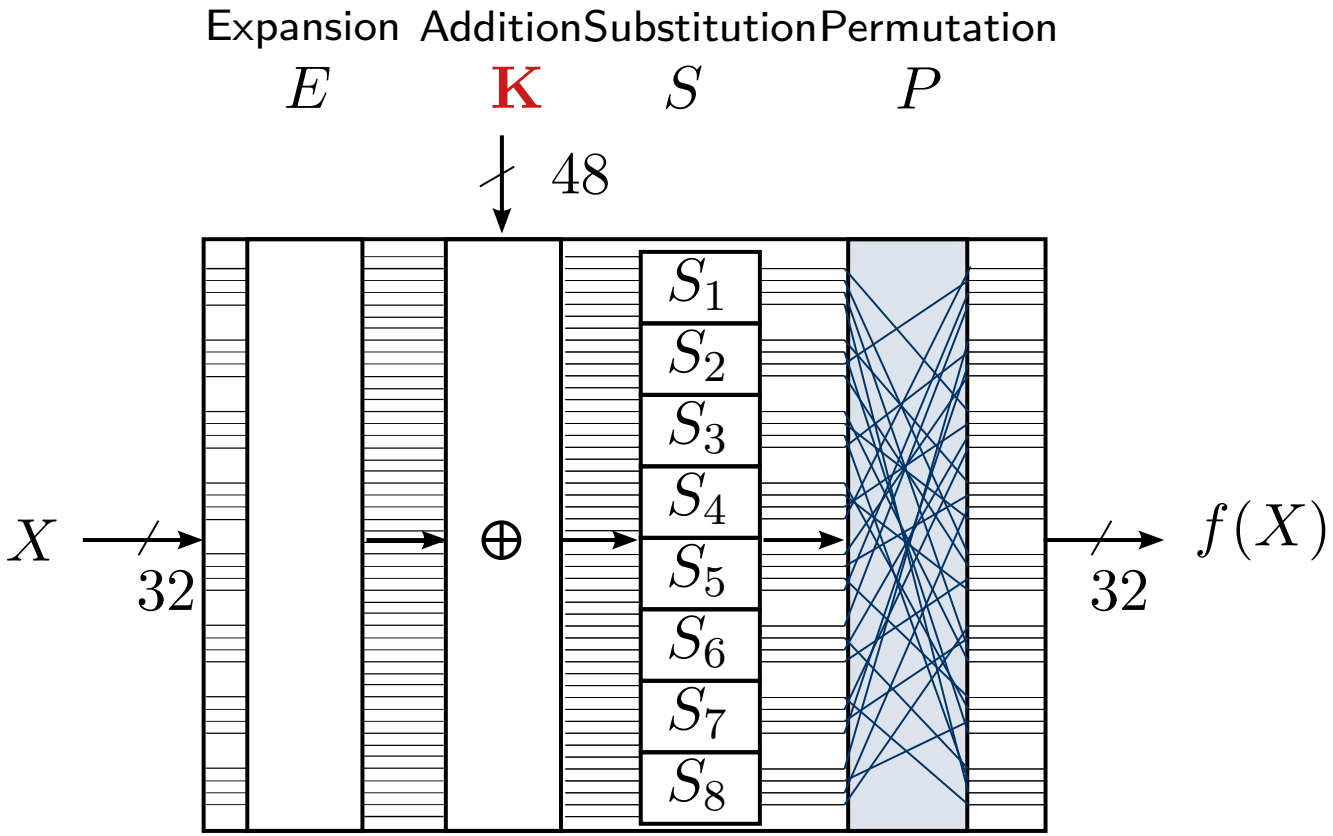
	Mittlere vier Bits der Eingabe															
S_1	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0 1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1 0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
1 1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	3

Beispiele

$$S_1(1) = S_1(\mathbf{000001}) = (0)_{10} = (0000)_2$$
$$S_1(20) = S_1(\mathbf{010100}) = (6)_{10} = (0110)_2$$
$$S_1(56) = S_1(\mathbf{111000}) = (3)_{10} = (0011)_2$$
$$S_1(57) = S_1(\mathbf{111001}) = (10)_{10} = (1010)_2$$

Die Permutation (P)

P-Permutation: 32 bit $\rightarrow$ 32 bit



Linearität

Linearität

Wir nennen eine Funktion $F : \mathcal{X} \to \mathcal{Y}$ **affin** (bzgl. einer Operation $\circ$) wenn für ein $\mathbf{A}$ und $B \neq 0$ und alle $X \in \mathcal{X}$ gilt

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

Wir nennen F **linear** wenn

$$F(X) = \mathbf{A} \cdot X.$$

- Wir beziehen uns hier auf Linearität bzgl. XOR (Addition in $\mathbb{GF}(2^n)$):

$$F(X) \oplus F(X') = F(X \oplus X').$$

Linearität der DES-Operationen

Bis auf die S-Boxen sind alle Operationen der DES-Rundenfunktion **linear** (bzgl. XOR):

- Unäre Operationen (E, P):

$$E(X_1) \oplus E(X_2) = E(X_1 \oplus X_2) \oplus E(0^{32})$$
$$P(X_1) \oplus P(X_2) = P(X_1 \oplus X_2) \oplus P(0^{32})$$

- Addition des Rundenschlüssels $KA_{\mathbf{K}[i]}(X) = X \oplus \mathbf{K}[i]$:

$$(X_1 \oplus \cancel{\mathbf{K}[i]}) \oplus (X_2 \oplus \cancel{\mathbf{K}[i]}) = X_1 \oplus X_2$$

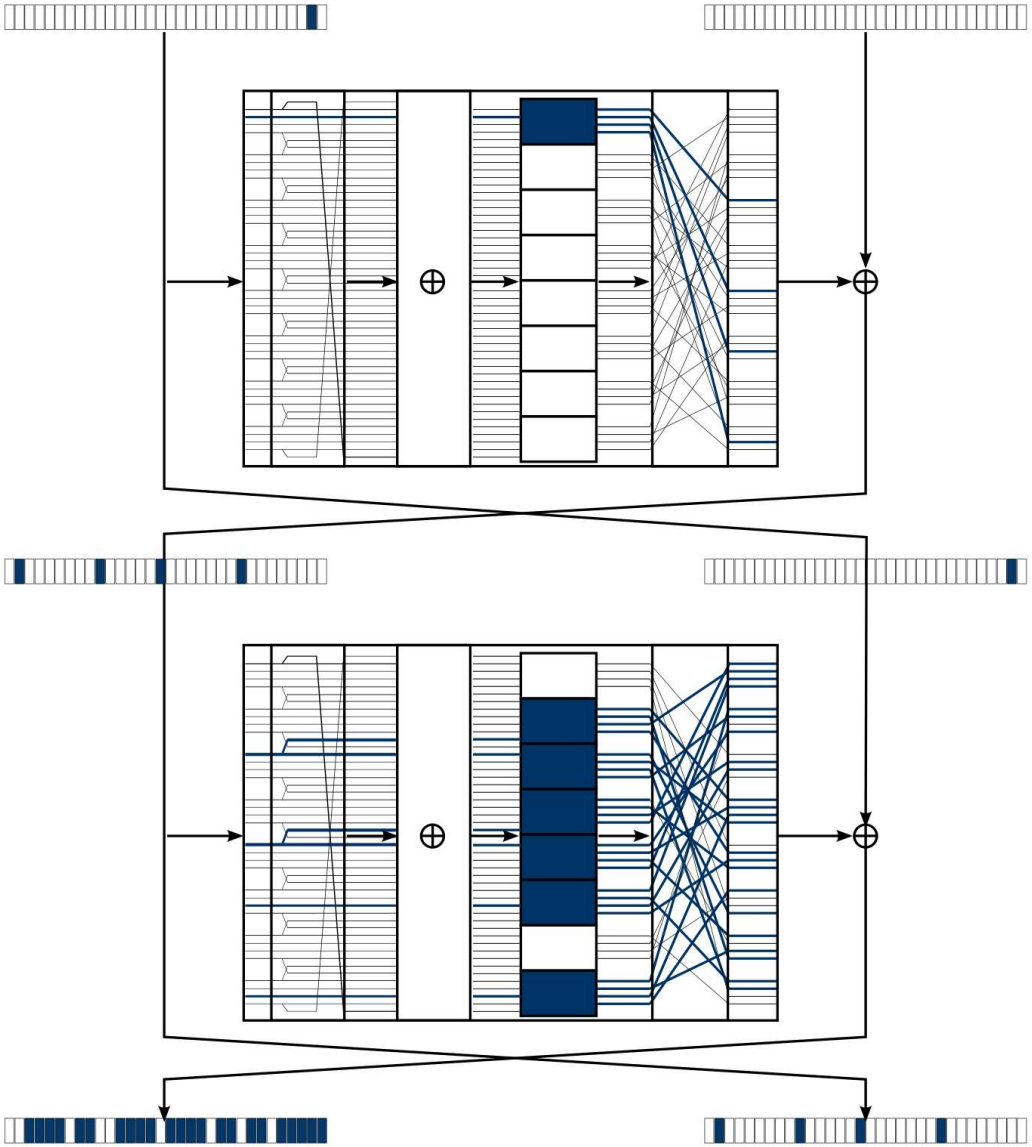
Das heißt, sei $f \in \{E, P, KA\}$:

- Sind X_1, $f(X_1)$ und die Änderung $X_2 \oplus X_1$ bekannt, so kann man $f(X_2)$ **ohne Kenntnis des Schlüssels** einfach berechnen.
- $\rightarrow$ eine Chiffre benötigt nicht-lineare Operation(en)!

Lawineneffekt (Diffusion)

- *Kleine* Änderungen in der Eingabe $\rightarrow$ *große* Änderung in der Ausgabe.
- Flippen eines Eingabebits $\rightarrow$ viele Ausgabebits können sich ändern.
- Flippen eines Eingabebits $\rightarrow$ jedes Ausgabebit kann sich ändern.
- Flippen eines Eingabebits $\rightarrow$ jedes Ausgabebit hat eine 50% Chance, sich zu ändern, unabhängig von den anderen Ausgabebits ("strict avalanche criterion", Webster, Tavers, 1985).

Beispiel Lawineneffekt: 2 Runden DES



- Beim DES verursacht die Kombination aus S-Boxen und der Permutation P den Lawineneffekt.
- Blau = Bits die sich ändern (können)

Komplementäreigenschaft

Sei $\overline{X}$ das Inverse des Bit-Strings X.

Theorem 15 (Komplementäreigenschaft)

Für alle Schlüssel $\mathbf{K}$ *und alle Klartexte* M *gilt*

$$\overline{DES_{\mathbf{K}}(M)} = DES_{\overline{\mathbf{K}}}(\overline{M}).$$

Angriffe auf den DES

Die wichtigten Angriffe auf den DES:

- Differentielle Kryptanalyse (demnächst)
- Lineare Kryptanalyse (werden wir nicht betrachten)
- Angriffe, die die kurze Schlüssellänge ausnutzen

Angriffe über die Schlüssellänge

Da DES-Schlüssel aus nur 56 bit bestehen, sind Brute-Force Angriffe mit der Rechenzeit $N = O(2^{56})$ durchaus praktikabel:

Vollst. Suche known plaintext, known ciphertext

Zeit $O(N)$, Platz $O(1)$

Tabellensuche chosen plaintext, known plaintext

Vorbereitungszeit $O(N)$, Platz $O(N)$,
Ausführungszeit $O(1)$

Time-Memory-Tradeoff (Hellman, 1980)

chosen plaintext, prinzipiell known plaintext

Vorbereitungszeit $O(N)$, Platz: $O(N^{2/3})$,
Ausführungszeit $O(N^{2/3})$

Geschichte:

1980 Hellman Time-Memory-Tradeoff
(Spezialrechner + Massenspeicher):
4 Mio. \$, 2 Jahre Vorbereitungszeit, 100 Schlüssel/Tag.

1993 Wiener (Spezialrechner):
1 Mio. \$, 7 Schlüssel/Tag.

1997 Erste DES-CHALLENGE
(Internet und *idle time* tausender Rechner):
keine Kosten, 4 Monate/Schlüssel.

1998 DES-Cracker der EFF (Spezialrechner):
250 000 \$, einige Tage/Schlüssel.

Vergleich: 1 Spionagesatelit 3 000 Mio. \$ bis 6 000 Mio. \$ (geschätzt).

Effektive Schlüssellänge

Eine Chiffre hat die **effektive Schlüssellänge** L bit, wenn es keinen Angriff gibt, der im Durchschnitt schneller ist als 2^{L-1} Verschlüsselungsoperationen. (Maßstab: Brute Force.)

Andere Ressourcen, insbesondere Speicherplatz und Klar-/Chiffretextpaare, können ebenfalls im Umfang bis zu 2^{L-1} Einheiten beansprucht werden.

Für praktikable Chiffren kennt man die effektive Schlüssellänge nicht. Man kennt nur obere Schranken ($\rightarrow$ Angriffe).

Folgerungen für den DES

- Der beste bekannte analytische Angriff (mittels linearer Kryptanalyse) braucht etwa 2^{43} bekannte Klar-Chiffretext-Paare.

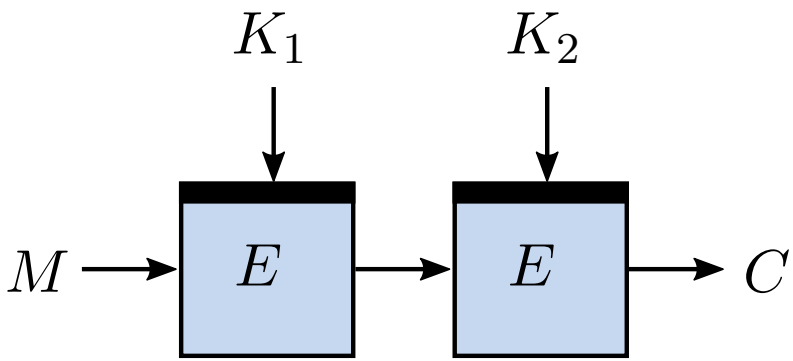
$\Rightarrow$ Effektive Schlüssellänge ≤ 44 bit.

- Alle bekannten analytischen Angriffe sind kaum praktikabel. Brute-Force-Angriffe sind praktikabel.

$\Rightarrow$ DES ist bemerkenswert stark gegen analytische Methoden, aber die Schlüssel sind zu klein.

Double-DES

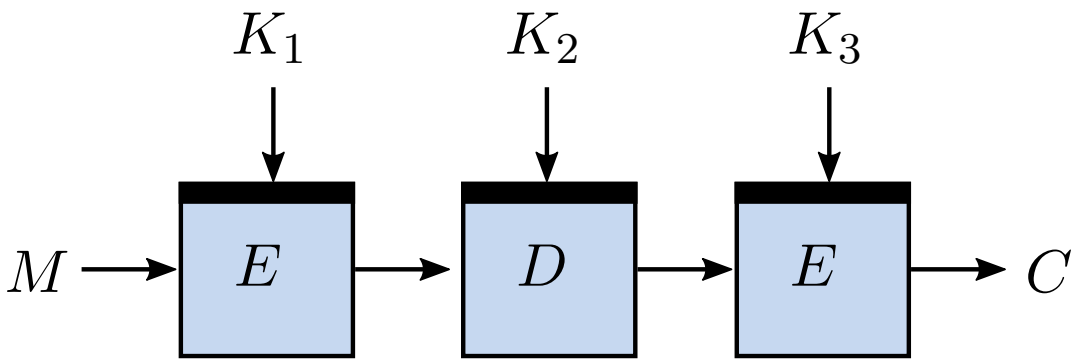
$$C = 2\mathrm{DES}_{\mathbf{K}_1,\mathbf{K}_2}(M) = \mathrm{DES}_{\mathbf{K}_2}\left(\mathrm{DES}_{\mathbf{K}_1}\left(M\right)\right)$$



Idee: Doppelte Anwendung von DES mit zwei unabhängigen Schlüsseln entspricht einem doppelt so großen Schlüssel, also 112 bit.
Stimmt das? ($\rightarrow$ Übung)

Triple-DES

$$3\mathrm{DES}_{\mathbf{K}_1,\mathbf{K}_2,\mathbf{K}_3}(M) := \mathrm{DES}_{\mathbf{K}_3}\left(\mathrm{DES}^{-1}_{\mathbf{K}_2}\left(\mathrm{DES}_{\mathbf{K}_1}(M)\right)\right)$$



Üblich: Statt der zweiten DES-Verschlüsselungsoperation eine DES-Entschlüsselungsoperation ("EDE"-Modus).

Angriffe auf Triple-DES

Variante	Angriff	# Paare	Rechenaufwand
Three-Key	MITM	3	2^{112}
Two-Key ($\mathbf{K}_1 = \mathbf{K}_3$)	[1]	2^{56}	2^{56}
Three-Key	[2]	2^{45}	2^{108}

[1] Merkle, Hellman (C. ACM, 1981).
[2] Lucks (FSE 1998).

DES: Zusammenfassung

- 64-bit-Blockchiffre
- 56-bit-Schlüssel
- bekannte und intensiv analysierte Blockchiffre
- massive Kritik an kurzen Schlüsseln
  Abhilfe: Triple DES
- Triple DES wird noch lange Zeit weiter genutzt werden
  (trotz des "DES-Nachfolgers" AES)

Sie sollten nun

- wissen, wie der DES funktioniert,
- einige Schwächen des DES kennen (schwache Schlüssel, . . .)
- diese Schwächen ggf. auch auf andere (ähnliche) Chiffren verallgemeinern können,
- und die Sicherheit von mehrfacher (doppelter, dreifacher, . . .) Verschlüsselung abschätzen können (mit Begründung).

4.3: Der Advanced Encryption Standard (AES)

- Problem: Schlüssellängen des DES nicht mehr ausreichend
- 1997: NIST beschließt öffentliche Ausschreibung um den AES
- Vorgaben:
  - 128-bit Blockchiffre
  - 3 Varianten: 128-bit, 192-bit und 256-bit Schlüssel
  - Sicher gegen alle bekannten Methoden der Kryptanalyse
  - Leicht in Hard- und Software zu implementieren
  - Schneller in Hard- und in Software als 3DES
  - Sicherer als 3DES
  - Patentfrei

Geschichte des AES (1)

1997 Ausschreibung des AES.

1998 1. AES-Konferenz; Präsentation von 15 Kandidaten.

"The Demolition Derby begins."

Feistel-Netzwerk			**SPN**		**Sonstige**
DES-ähnlich		Erweitert	Allgemein	SQUARE-ähnlich	
DEAL	DFC	Cast-256	SAFER+	Crypton	Frog
Loki97	E2	MARS	Serpent	Rijndael	HPC
Magenta	RC6				
Twofish					

Geschichte des AES (2)

1999 2. AES-Konferenz.

Angriffe DEAL, Frog, HPC, Loki97, Magenta.

Finalisten **MARS, RC6, Rijndael, Serpent, Twofish**.

Feistel-Netzwerk			**SPN**		**Sonstige**
DES-ähnlich		Erweitert	Allgemein	SQUARE-ähnlich	
DEAL	DFC	Cast-256	SAFER+	Crypton	Frog
Loki97	E2	**MARS**	**Serpent**	**Rijndael**	HPC
Magenta	**RC6**				
Twofish					

Geschichte des AES (3)

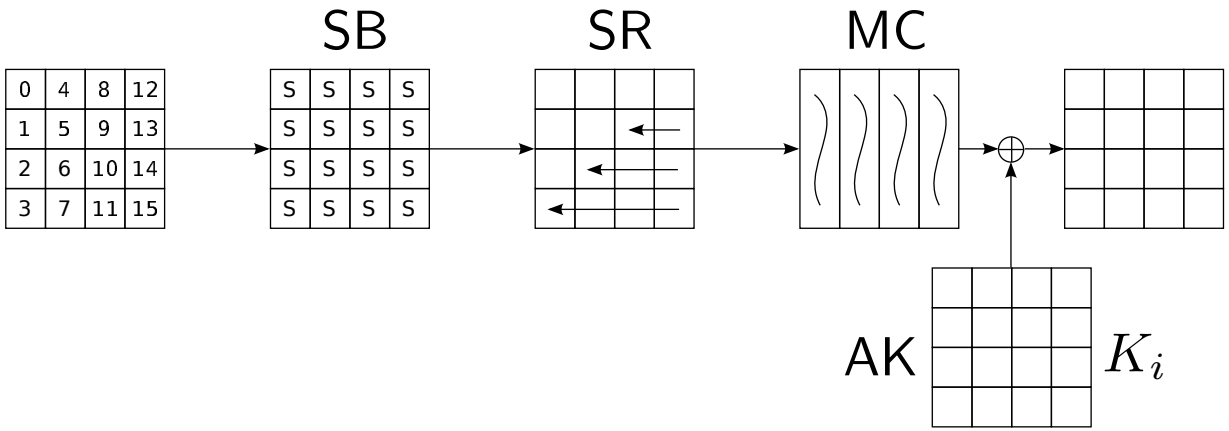
Apr 2000 3. AES-Konferenz, Diskussion der Finalisten.

Okt 2000 **Rijndael** (angepasst) wird vorläufiger Standard.

Nov 2001 NIST-Standard FIPS 197 verabschiedet.

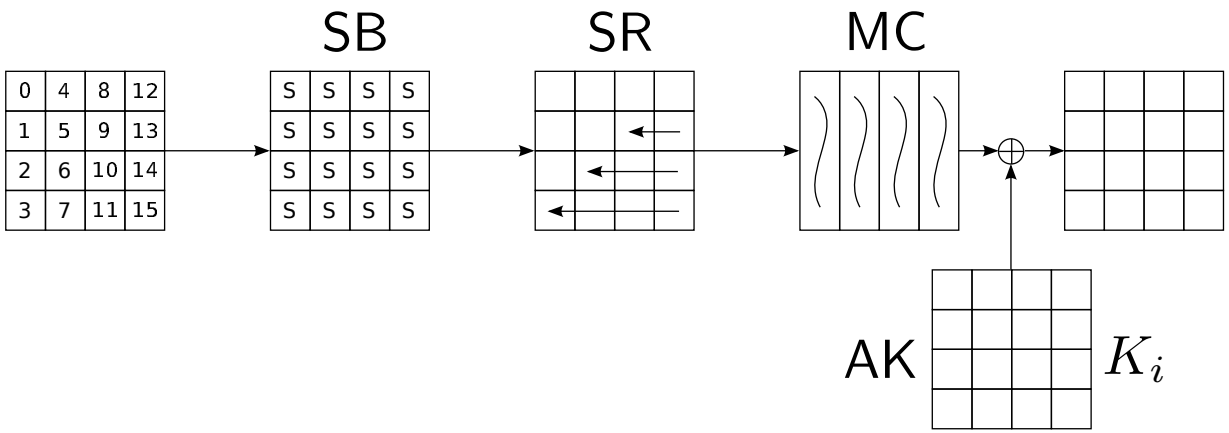
Feistel-Netzwerk			**SPN**		**Sonstige**
DES-ähnlich		Erweitert	Allgemein	SQUARE-ähnlich	
DEAL	DFC	Cast-256	SAFER+	Crypton	Frog
Loki97	E2	MARS	Serpent	**Rijndael**	HPC
Magenta	RC6				
Twofish					

Struktur des AES (von Rijndael)



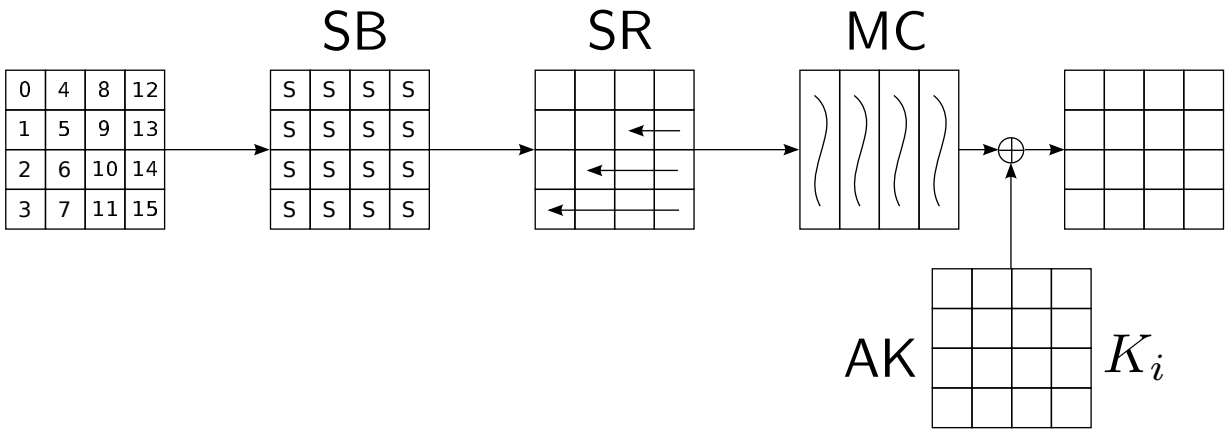
- Vier Basis-Operationen (SubBytes, ShiftRows, MixColumns, AddRoundKey)
- Eine AES-Runde als Kombination der vier Basis-Operationen
- Der "Key Schedule": Aus einem kurzen Chiffrier-Schlüssel (128 bit, 192 bit, 256 bit) werden 11 bis 15 Rundenschlüssel (jeweils 128 bit).
- 10 (für 128-), 12 (für 192-), 14 (für 256-bit-Schlüssel)
- Anzahl Rundenschlüssel $= 1 +$ Anzahl Runden

Die Struktur des AES



- 128-bit-Zustand, 4×4-Byte-Matrix
- 128-, 192-, 256-Bit-Schlüssel
- Rundenbasiertes Substitution-Permutations-Netzwerk
- 10, 12, 14 Runden mit den Operationen:
- Zu Beginn: AddRoundKey mit initialem Schlüssel K_0 (insgesamt 11/13/15 Rundenschlüssel)

Die Rundenoperationen des AES



SubBytes Tauscht jedes Byte mit Hilfe einer 8×8-Bit S-box aus.

ShiftRows Rotiert die i-te Zeile um $i \in \{0, 1, 2, 3\}$ Bytes nach links.

MixColumns Multipliziert jede Spalte mit einer 4×4-MDS-Matrix.

AddRoundKey XORe Rundenschlüssel K_i auf den Zustand

Die letzte Runde enthält *keine* MixColumns-Operation

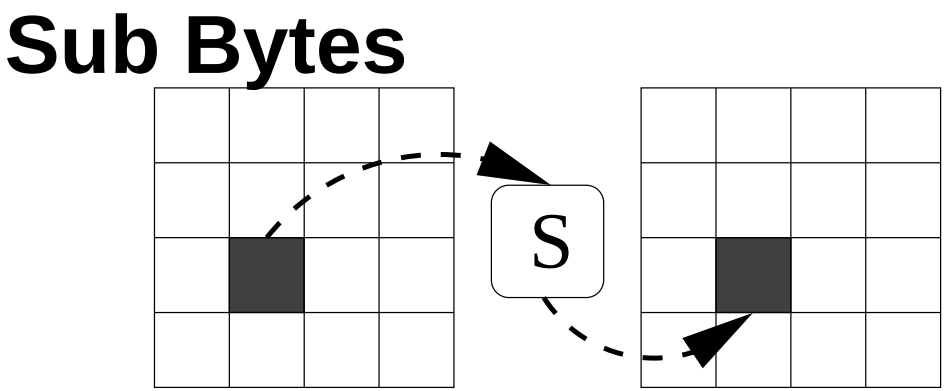
Die Rundenoperationen des AES

SubBytes

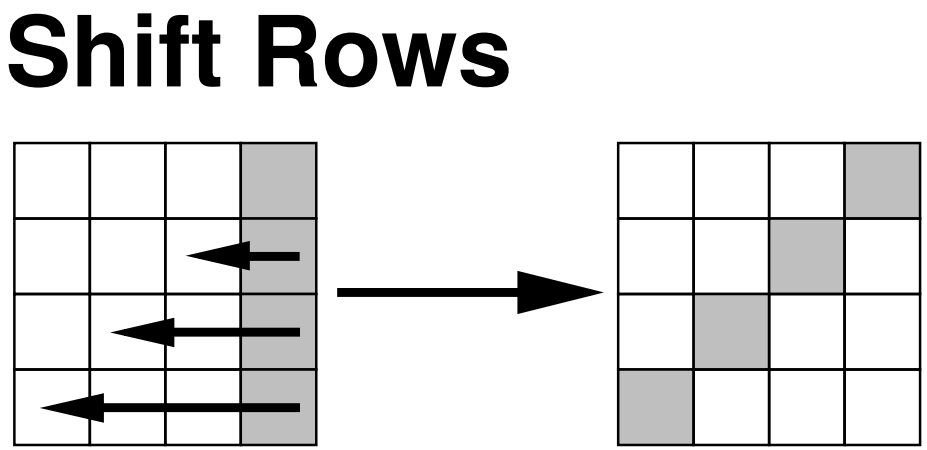
- Anwendung einer invertierbaren S-Box (Tabelle mit 2^8 Einträgen)

$$S : \{0,1\}^8 \to \{0,1\}^8$$

  mit einer sehr einfachen algebraischen Struktur.
- Trotz der einfachen algebraischen Struktur ist S eine nichtlineare Funktion.

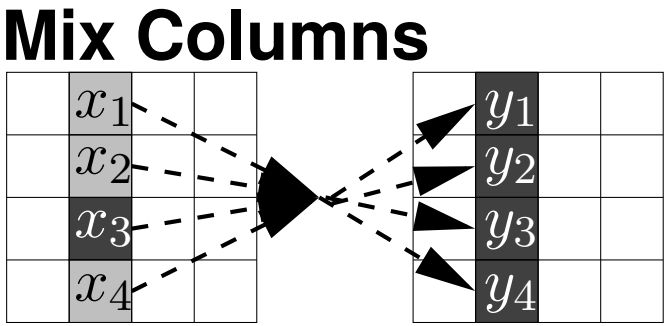


ShiftRows



MixColumns

- Interpretiert jede Spalte als Vektor $\mathbf{x} = (x_1, x_2, x_3, x_4)^T$
- MixColumns ist eine Matrix-Vektor-Multiplikation $\mathbf{y} = \mathbf{A} \cdot \mathbf{x}$ (im Galois-Körper $\mathbb{GF}(2^8)/p(x)$ mit $p(x) = \mathbf{x}^8 + \mathbf{x}^4 + \mathbf{x}^3 + \mathbf{x} + 1$)
- Da Matrix $\mathbf{A}$ invertierbar ist $\rightarrow$ MixColumns ist invertierbar



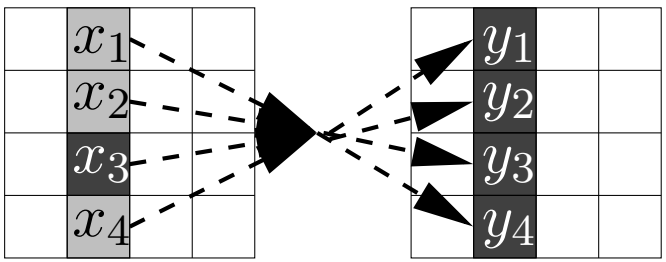
$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

MixColumns: Branch Number

Mindestanzahl > 0 der geänderten S-Box Eingaben in zwei aufeinanderfolgenden Runden

- MixColumns: Branch Number des AES $\mathcal{B} = 5$
- Eine Differenz in *genau einem* der Eingabe-Werte x_i führt zu einer Differenz in **allen vier** Ausgabewerten y_1, y_2, y_3, y_4!
- Eine Differenz in *genau* $d > 0$ der Eingabe-Werte x_i führt zu einer Differenz in **mindestens** $5 - d$ Ausgabewerten y_1, y_2, y_3, y_4

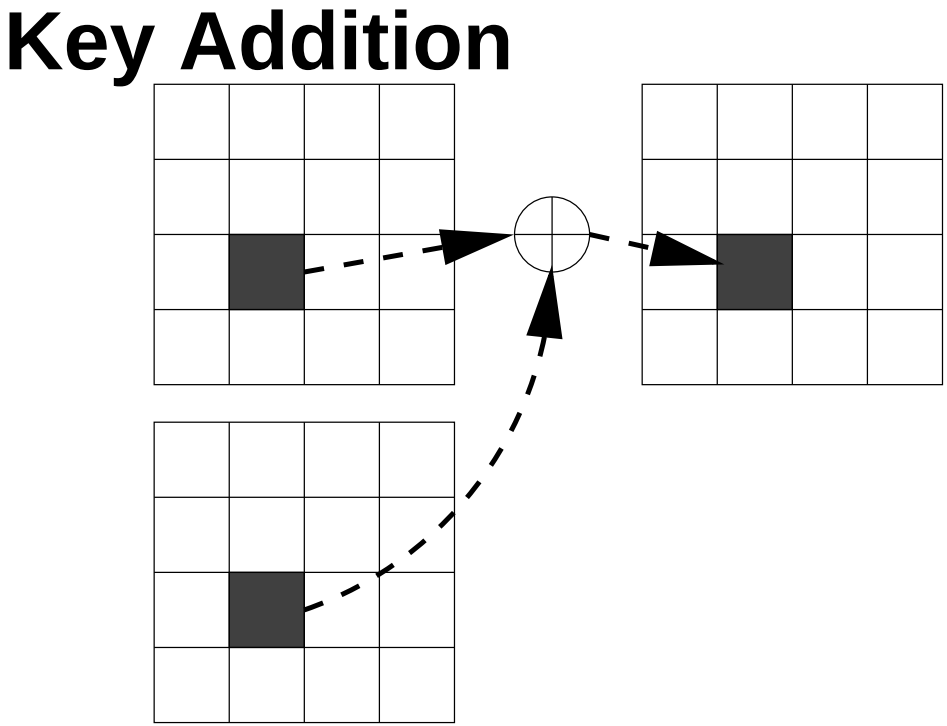
Mix Columns



$$\begin{aligned} y_1 &= 2x_1 + 3x_2 + x_3 + x_4 \\ y_2 &= x_1 + 2x_2 + 3x_3 + x_4 \\ y_3 &= x_1 + x_2 + 2x_3 + 3x_4 \\ y_4 &= 3x_1 + x_2 + x_3 + 2x_4 \end{aligned}$$

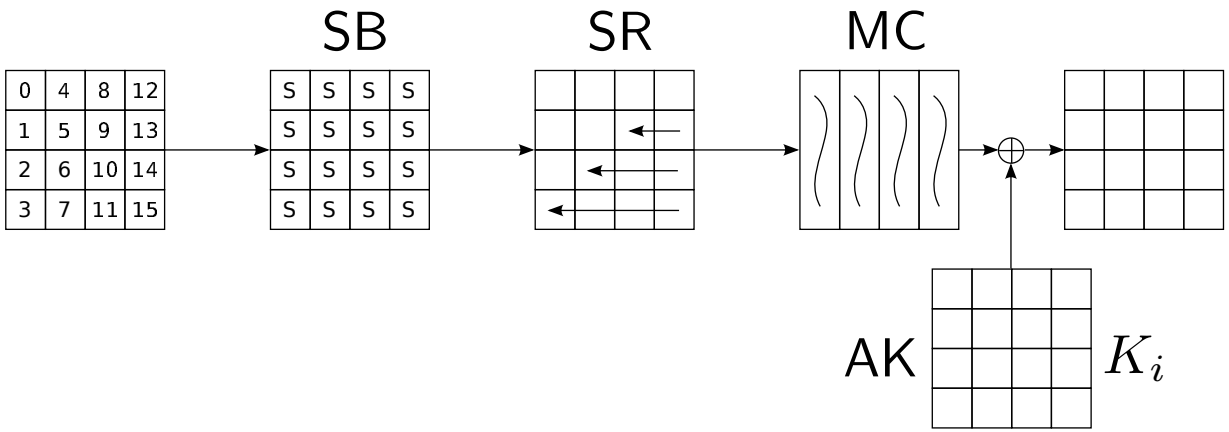
AddRoundKey

- Chiffrierschlüssel: k bit, $k \in \{128, 192, 256\}$
- Rundenschlüssel: $n + 1$ für n Runden, jeweils 128 bit
- Vor der ersten und nach jeder Runde:
  Addition (bitweise mod 2) eines Rundenschlüssel



Die Rundenoperationen des AES

Designentscheidungen



SubBytes Nicht-lineare Operation. Ohne sie:

$$\mathrm{AES}(x) \oplus \mathrm{AES}(y) = \mathrm{AES}(x \oplus y) \oplus \mathrm{AES}(0)$$

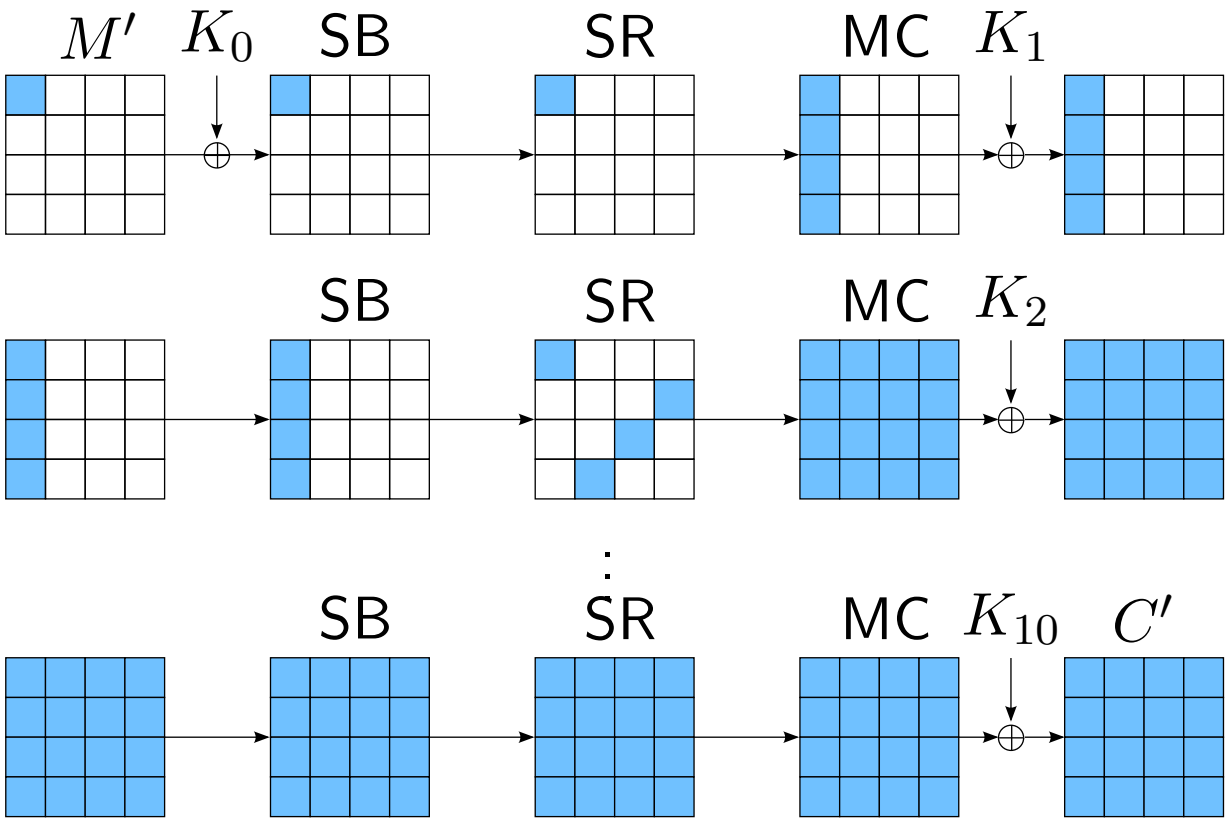
ShiftRows Verteile eine Differenz über mehrere Spalten.

MixColumns Verteile eine Differenz über mehrere Zeilen.

AddRoundKey Abhängigkeit vom Rundenschlüssel.

Die Rundenoperationen des AES

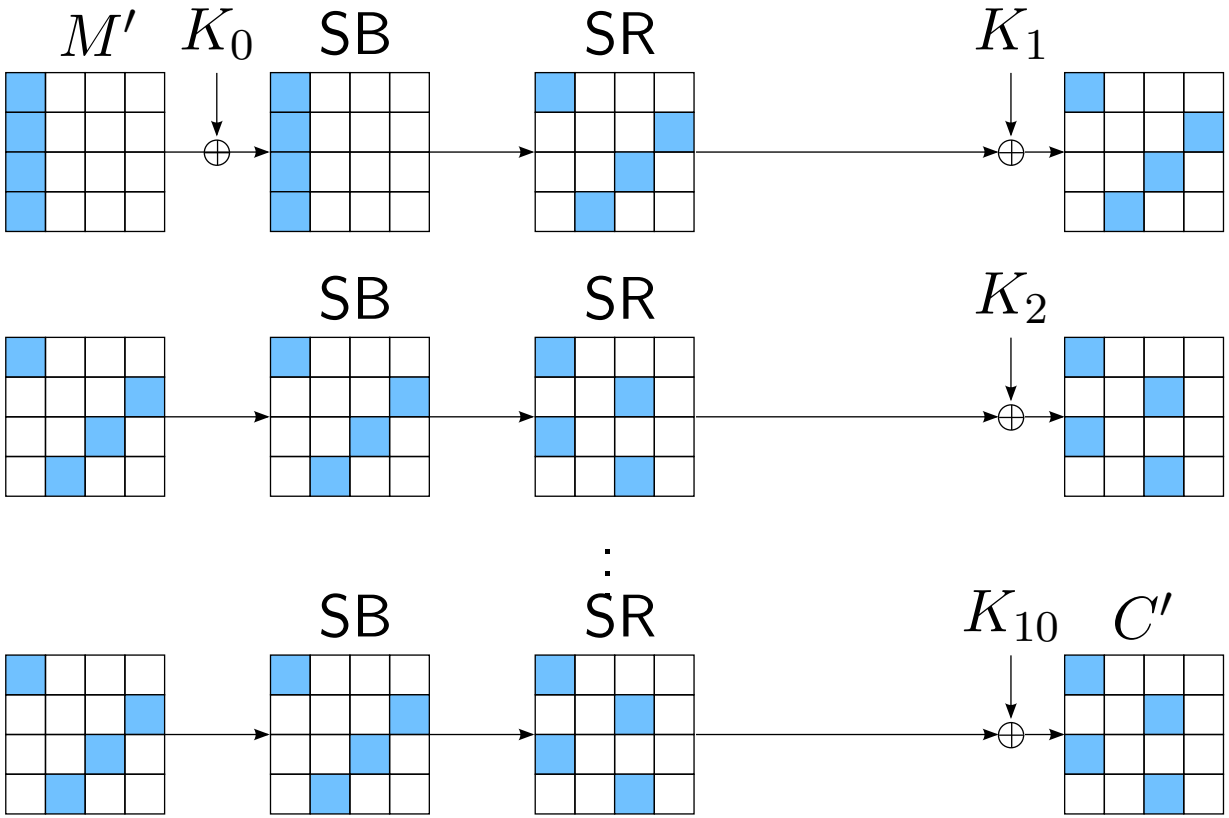
Diffusion im vollen AES



- Angenommen, wir kennen M und $C = AES_K(M)$
- Blau = wir ändern ein Byte in M für M'.

Die Rundenoperationen des AES

Diffusion ohne MixColumns

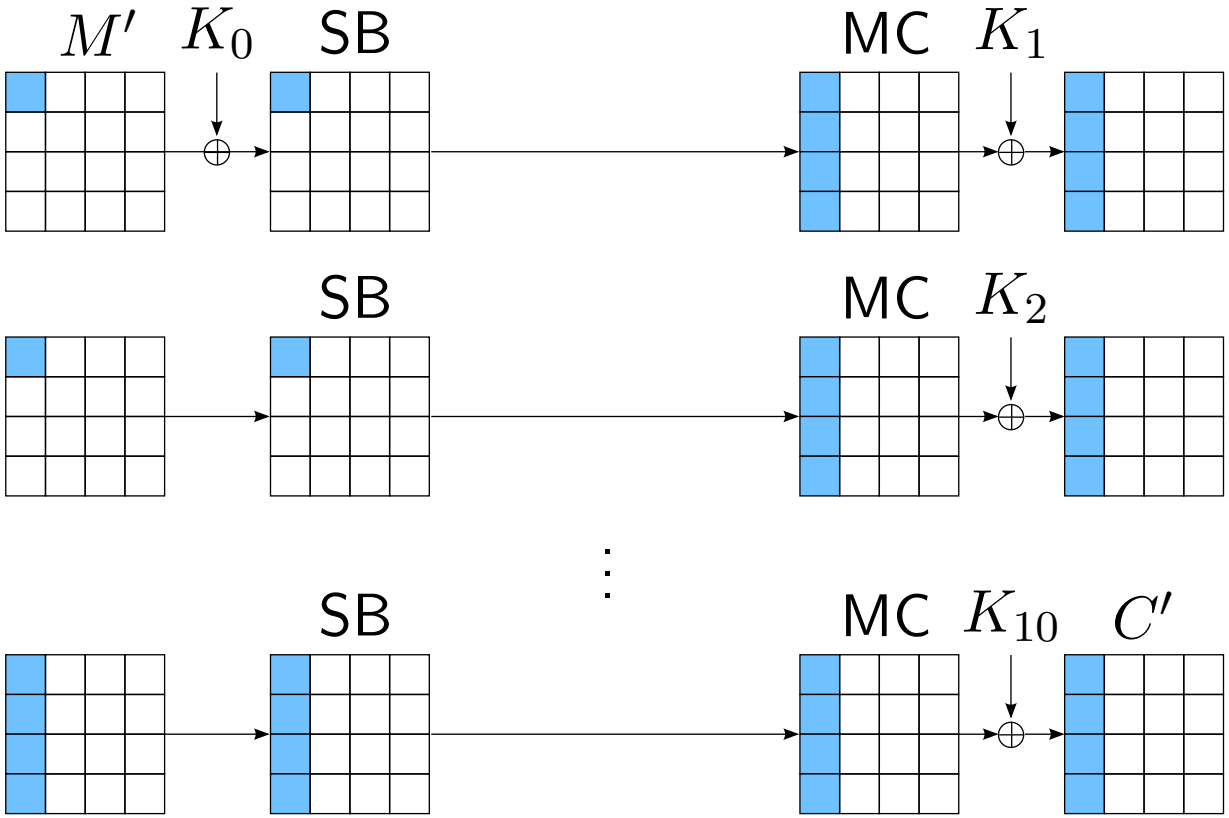


- Angenommen, die MixColumns-Operation würde fehlen

→ Differenz rotiert nur innerhalb der Zeilen

Die Rundenoperationen des AES

Diffusion ohne ShiftRows



- Angenommen, die ShiftRows-Operation würde fehlen

$\rightarrow$ Keine Diffusion über Spaltengrenzen

AES: Zusammenfassung

- 128-bit-Blockchiffre
- 3 verschiedene Schlüssellängen: 128 bit, 192 bit, 256 bit
- "Rijndael" ging als Sieger aus einem mehrjährigen internationalen Wettbewerb hervor
- Aktueller Standard
- Inzwischen ähnlich intensiv analysiert wie der DES

Sie sollten grundlegende Vorstellungen davon haben,

- wie der AES funktioniert
- und warum die Struktur des AES so ist, wie sie ist (was passiert, z.B., wenn man eine der 4 Grundoperationen weglässt?).