

Alle Angaben ohne Gewähr. Keine Garantie auf Vollständigkeit oder Richtigkeit.

1	Einführung	2
1.1	Ausgangspunkte für Angriffe	2
1.2	Angriffsarten	2
1.3	Historische Verschlüsselungsverfahren	2
2	Blockchiffren	2
2.1	Definitionen	2
2.1.1	Definition: Blockchiffre	2
2.1.2	Anforderungen an Blockchiffren	2
2.1.3	Definition: Ideal Cipher	2
2.1.4	Anforderungen an Ideal Cipher	3
2.2	DES (Data Encryption Standard)	3
2.2.1	Beispiel für Encryption-Schritt	3
2.2.2	Beispiel für Decryption-Schritt	3
2.2.3	F-Funktion	3
2.3	2DES	4
2.3.1	Angriff auf 2DES	4
2.4	3DES	4
2.4.1	Aufwand für Meet-in-the-Middle-Angriffe gegen 3DES	4
2.4.2	2Key-3DES	5
2.4.3	Advanced Meet-in-the-Middle-Angriff gegen 2Key-3DES	5
2.4.4	Aufwand für Advanced Meet-in-the-Middle-Angriffe gegen 2Key-3DES	5
2.5	Slide-Attacks	5
2.5.1	Zellularautomaten	5
2.5.2	Verschlüsselung mit Zellularautomaten	6
2.5.3	Slide-Attack auf Zellularautomaten	6
2.5.4	Advanced Slide-Attack gegen DES-X	7
2.5.5	Aufwand für Advanced Slide-Attack gegen DES-X	8
3	Lineare Kryptoanalyse	8
3.1	FEAL-4	8
3.1.1	F-Funktion	8
3.1.2	Beobachtung	9
3.1.3	Charakteristik	9
3.1.4	Angriff	10
3.1.5	Aufwand	10
3.2	DES	10
3.2.1	Angriff	10
3.2.2	Aufwand	10
4	Differentielle Kryptoanalyse	10
4.1	DES	10
4.1.1	Idee	10
4.1.2	Angriff auf 1-Runden-DES	11

1 Einführung

1.1 Ausgangspunkte für Angriffe

Angriffe können nach den zur Verfügung stehenden Informationen unterteilt werden:

- **Ciphertext-Only-Attack**: Nur das *Chiffre*, also die verschlüsselte Nachricht, ist bekannt
- **Known-Plaintext-Attack**: Es gibt bekannte Klartext-Chiffre-Paare. Hilfreich sind bekannte Anfangs- und Endphrasen, die in mehreren Nachrichten vorkommen.
- **Chosen-Plaintext-Attack**: Es besteht die Möglichkeit, beliebige Texte zu verschlüsseln und somit Klartext-Chiffre-Paare zu erzeugen.

1.2 Angriffsarten

- Brute-Force (z.B. alle Schlüssel ausprobieren)
- Statistische Methoden (z.B. Häufigkeitsanalysen von Buchstaben)
- Strukturelle Angriffe (z.B. Lineare Kryptoanalyse)

1.3 Historische Verschlüsselungsverfahren

Historisch wurden zur Verschlüsselung zwei grundlegende Operationen verwendet:

- **Substitution**
- **Permutation**

Alleine sind beide Verfahren meistens nicht sicher, jedoch verwenden moderne Verschlüsselungsverfahren eine Kombination beider Operationen.

2 Blockchiffren

2.1 Definitionen

2.1.1 Definition: Blockchiffre

Gegeben seien zwei endliche Alphabete A, B und $n, m \in \mathbb{N}$ sowie ein Schlüsselraum \mathcal{K} . Eine **Blockchiffre** ist gegeben durch eine Familie von injektiven Abbildungen $f_k : A^n \rightarrow B^m$ mit $k \in \mathcal{K}$. In der Regel gilt $A = B = \{0, 1\}$ und $n = m$.

2.1.2 Anforderungen an Blockchiffren

- Gegeben den Schlüssel k müssen sowohl f_k als auch f_k^{-1} **effizient** berechenbar sein
- Ein Angreifer soll nicht zwischen einer *zufälligen Abbildung* und der Blockchiffre mit *zufälligem Schlüssel* unterscheiden können

2.1.3 Definition: Ideal Cipher

Eine **Ideal Cipher** (IC) ist eine (Über-)Idealisierung einer Blockchiffre. Jedem Schlüssel $k \in \{0, 1\}^\lambda$ ist eine vollkommen zufällige Permutation $P_k : \{0, 1\}^n \rightarrow \{0, 1\}^n$ zugeordnet (hierbei sind λ und n Sicherheitsparameter) und per Orakelzugriff kann jede Maschine im Modell die Funktionen P_k und P_k^{-1} auswerten. Die Existenz einer solchen IC wird zur Vereinfachung von Beweisen angenommen, man spricht dann von dem **Ideal-Cipher-Modell**.

2.1.4 Anforderungen an Ideal Cipher

- Alle Parteien können über Orakelzugriff P_k und P_k^{-1} auswerten
- Ideal Cipher liefert zu jedem Paar (k, m) ein c "zufällig" gewählt
- Ideal Cipher liefert zu jedem Paar (k, c) ein m "zufällig" gewählt
- Orakel muss jede Ausgabe speichern, damit für gleiche Nachrichten immer das gleiche Chiffre zurückgegeben wird (nicht parallelisierbar)

2.2 DES (Data Encryption Standard)

Der **Data Encryption Standard** ist eine Blockchiffre mit Schlüssellänge $k = 56$ und Blocklänge $n = 64$, die Verschlüsselungsfunktion ist also $\{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$. Er besteht aus einer **Feistel-Struktur** mit 16 Runden und wurden aufgrund der kurzen Schlüssellänge **gebrochen**.

2.2.1 Beispiel für Encryption-Schritt

$$L_1 = R_0$$

$$R_1 = L_0 \oplus F_{k_1}(R_0)$$

$$L_{16} = R_{15}$$

$$R_{16} = L_{15} \oplus F_{k_{16}}(R_{15})$$

2.2.2 Beispiel für Decryption-Schritt

$$R_{15} = L_{16}$$

$$L_{15} = R_{16} \oplus F_{k_{16}}(R_{15})$$

$$= R_{16} \oplus F_{k_{16}}(L_{16})$$

2.2.3 F-Funktion

Die **F-Funktion** ist eine nicht-reversible Funktion, die in jeder Runde der Feistel-Struktur ausgeführt wird. Der Ablauf ist folgender:

1. **Expansion:** Die 32 Eingabebits werden auf 48 Bits erweitert
2. Das bitweise XOR zwischen Expansion und dem Schlüssel wird berechnet
3. Das Ergebnis wird in 6-Bit-Blöcken auf 8 **Substitutionsboxen** verteilt
4. Die Substitution wird permutiert und ausgegeben

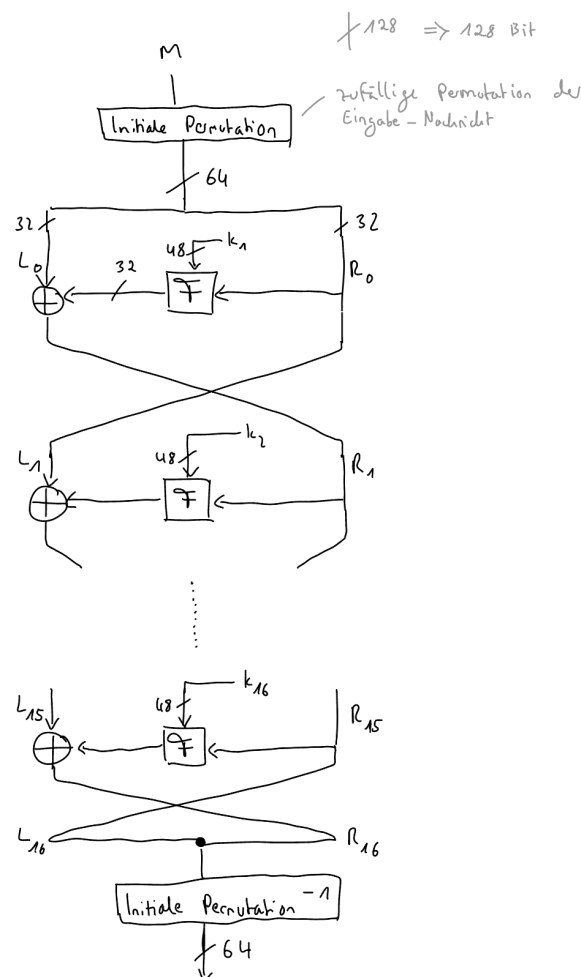


Abbildung 1: DES-Verschlüsselungsalgorithmus

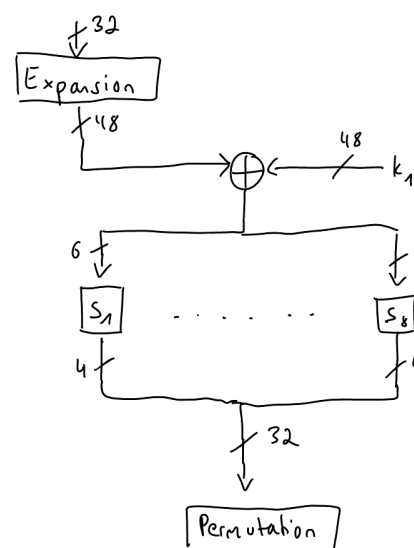


Abbildung 2: F-Funktion

2.3 2DES

DES wurde aufgrund der kurzen Schlüssellänge (56 Bit) gebrochen und sollte daher nicht mehr in seiner einfachen Form verwendet werden. Es gibt jedoch modifizierte Varianten, die die Sicherheit von DES verbessern, z.B. **2DES**:

Bei 2DES wird die Nachricht zuerst mit k_1 verschlüsselt und das Chiffre dann erneut mit k_2 verschlüsselt:
$$c = DES_{k_2}(DES_{k_1}(m))$$

2.3.1 Angriff auf 2DES

Gegen 2DES sind **Meet-in-the-Middle**-Angriffe möglich, dabei wird versucht, die Verschlüsselung von beiden Seiten zu brechen.

Gegeben sind zwei Paare (m_1, c_1) und (m_2, c_2) , dann funktioniert der Angriff wie folgt:

1. **Vorwärtsschritt**: Berechne $DES_k(m_1)$ und $DES_k(m_2)$ für alle $k = 0, \dots, 2^{56} - 1$ und speichere alle Werte in einer Tabelle T
2. **Sortierschritt**: Sortiere Tabelle T
3. **Rückwärtsschritt**: Berechne $DES_k^{-1}(c_1)$ und $DES_k^{-1}(c_2)$ für alle $k = 0, \dots, 2^{56} - 1$ und suche nach Treffern in T

Zeitaufwand des Angriffs:

1. **Vorwärtsschritt**: $2 * 2^{56}$ DES-Operationen
2. **Sortierschritt**: $56 * 2^{56}$ Vergleiche
3. **Rückwärtsschritt**: $2 * 2^{56}$ DES-Operationen (da man nach ungefähr nach der Hälfte fertig ist)

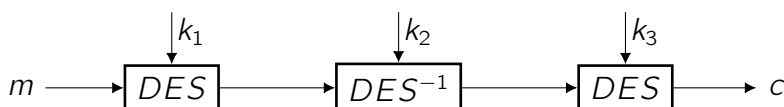
Speicheraufwand des Angriffs:

In der Tabelle müssen ungefähr 2^{60} Byte gespeichert werden, was ungefähr 1.150.000 TB entspricht. Damit ist der Angriff so nicht praktikabel.

Der Angriff lässt sich "verbessern", indem ein Teil des Platzbedarfs mit erhöhtem Rechenaufwand ausgeglichen wird, somit ist das Speicherproblem nicht mehr unüberwindbar, der Angriff dauert aber proportional länger.

2.4 3DES

3DES ist analog zu 2DES definiert. Hierbei werden drei verschiedene Schlüssel k_1, k_2, k_3 verwendet. In der mittleren Box wird statt dem normalen DES die inverse Funktion DES^{-1} verwendet. Dadurch kann bei Bedarf $k_1 = k_2 = k_3$ gesetzt werden, wodurch effektiv nur eine DES -Verschlüsselung mit k_1 ausgeführt wird.

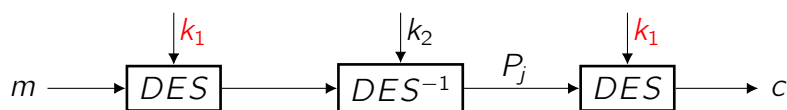


2.4.1 Aufwand für Meet-in-the-Middle-Angriffe gegen 3DES

- **Zeit** $\approx 2^{112}$
- **Platz** $\approx 2^{56}$

2.4.2 2Key-3DES

2Key-3DES ist eine Variation von 3DES, das einen Schlüssel wiederverwendet:



2.4.3 Advanced Meet-in-the-Middle-Angriff gegen 2Key-3DES

1. Wähle $0 \dots 0$ als Ergebnis nach erster DES-Box, entschlüssele für alle möglichen Schlüssel: $DES_k^{-1}(0 \dots 0)$ und schreibe Ergebnis in eine Tabelle (die Berechnung liefert sowohl m als auch P_j)
2. Lasse alle 2^{56} Klartexte m verschlüsseln (chosen-plaintext-attack)
3. Entschlüssele jedes Chiffre mit **dem einen** bekannten k_1
4. Suche den Wert in der Tabelle, bei Treffern *Kandidat* für (k_1, k_2)
5. Überprüfe an weiteren (m, c) -Paaren

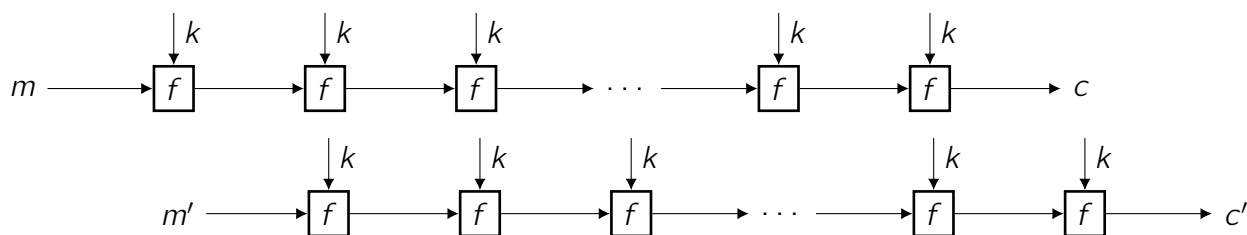
2.4.4 Aufwand für Advanced Meet-in-the-Middle-Angriffe gegen 2Key-3DES

- **Zeit** $\approx 3 * 2^{56}$
- **Platz** $\approx 2^{56}$

2.5 Slide-Attacks

Slide-Attacks sind eine besondere Art von Angriffen, die nur bei Verschlüsselungsverfahren mit besonderer Struktur funktionieren. Die Verschlüsselung muss in mehreren Runden geschehen, wobei in jeder Runde die **gleiche Funktion** mit **gleichem Schlüssel** zum verwendet wird.

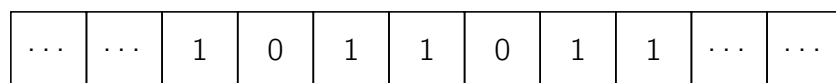
Findet man nun zwei Paare (m, c) und (m', c') wie in der folgenden Grafik, muss nur noch eine Runde gebrochen werden, um den Schlüssel zu erhalten:



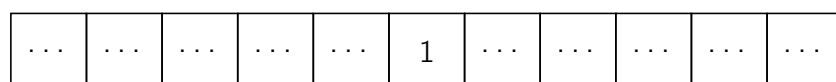
2.5.1 Zellularautomaten

Eindimensionale, binäre Zellularautomaten bestehen aus n Zellen, die jeweils ein Bit enthalten.

Die Übergangsfunktion f erhält typischerweise k (z.B. 5) Nachbarn (inklusive der Zelle selbst) als Eingabe und gibt als Ergebnis das Bit aus, das im nächsten Schritt an dieser Position steht:



Zeitpunkt t



Zeitpunkt $t + 1$

Die Übergangsfunktion kann auch als Tabelle dargestellt werden:

Nachbarschaft N	$f(N)$
(0, 0, 0, 0, 0)	0
\vdots	\vdots
(0, 1, 1, 0, 1)	1
\vdots	\vdots
(1, 1, 1, 1, 1)	0

2.5.2 Verschlüsselung mit Zellularautomaten

Um Zellularautomaten als Verschlüsselungsverfahren zu verwenden, müssen sie zudem invertierbar sein, was standardmäßig nicht gegeben ist. Dafür verwenden wir eine Feistel-Struktur, bei der wir das Ergebnis $f(N)$ mit dem bisherigen Wert an dieser Stelle XORen:

...	1
-----	-----	-----	-----	-----	---	-----	-----	-----	-----	-----

Zeitpunkt 0: IV

...	...	1	0	1	1	0	1	1
-----	-----	---	---	---	---	---	---	---	-----	-----

Zeitpunkt 1: m

1

...	0
-----	-----	-----	-----	-----	---	-----	-----	-----	-----	-----

Zeitpunkt 2

\vdots

...
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Zeitpunkt n : c

...
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Zeitpunkt $n + 1$: Zusatzinformation

Die Zusatzinformation wird benötigt, um zu entschlüsseln.

2.5.3 Slide-Attack auf Zellularautomaten

Bei dem Angriff handelt es sich um einen **Chosen-Plaintext-Angriff**, der wie folgt funktioniert:

1. Setze $m := 1 \dots 1$ und $IV := 0 \dots 010 \dots 0$ (IV überall 0, außer an einer Stelle)
2. In Zeitpunkt 2 steht damit überall der selbe Wert $a \in \{0, 1\}$, außer an einer Stelle, wo $\neg a$ steht
3. Speichere m , c und ZI (Teil 1)
4. Verschlüssele nun $m' = a \dots a \neg a a \dots a$ (für $a = 0$ und $a = 1$)
5. Bei einem der beiden Fälle erhalten wir als Chiffre c' (gleich dem Chiffre aus Teil 1) und ZI'
6. Da wir zusätzlich zur Übergangsfunktion XORen, machen wir diese Operation mit $b = c \oplus ZI'$ rückgängig
7. b ist dann das Ergebnis der Übergangsfunktion zum Zeitpunkt $n + 1$
8. Damit haben wir Eingaben c' und Ergebnisse b der Übergangsfunktion und können anfangen, die Tabelle zu befüllen
9. Wiederhole das Vorgehen mit anderen IV , bis die Tabelle vollständig ist
10. Die fertige Tabelle ist der gesuchte Schlüssel

Visualisierung:

0	0	0	0	0	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
a	a	a	a	a	¬a	a	a	a	a	a
⋮										
...
...
...

Zeitpunkt 0: IV

Zeitpunkt 1: m

Zeitpunkt 2: m'

Zeitpunkt n : c

Zeitpunkt $n+1$: $ZI = c'$

Zeitpunkt $n+2$: ZI'

2.5.4 Advanced Slide-Attack gegen DES-X

DES-X verwendet DES als Grundlage, arbeitet jedoch zwei Schlüsseln k_y, k_x (jeweils 64-Bit):

$$c = \text{DES-X}(m) = k_y \oplus \text{DES}_k(m \oplus k_x)$$

$$m = \text{DES-X}^{-1}(c) = k_x \oplus \text{DES}_k^{-1}(c \oplus k_y)$$

Auf den ersten Blick ist nicht klar, wie eine Slide-Attack auf DES-X funktionieren kann, da nichts mehrfach durchgeführt wird. Werden Ver- und Entschlüsselung leicht überlappend aneinandergehangen, kann allerdings ein Angriff konstruiert werden.

Was suchen wir überhaupt?

- Wir benötigen "slid pairs" (m, c) und (m', c') mit $c \oplus c' = k_y$
- m erhalten wir durch Zurückrechnen von DES:

$$\begin{aligned} m &= k_x \oplus \text{DES}_k^{-1}(c \oplus k_y) \\ &= k_x \oplus \text{DES}_k^{-1}(c') \end{aligned}$$

- m' erhalten wir analog:

$$\begin{aligned} m' &= k_x \oplus \text{DES}_k^{-1}(c' \oplus k_y) \\ &= k_x \oplus \text{DES}_k^{-1}(c) \end{aligned}$$

- Durch Auflösen nach k_x und Gleichsetzen ergibt sich:

$$\begin{aligned} m \oplus \text{DES}_k^{-1}(c') &= m' \oplus \text{DES}_k^{-1}(c) \\ \Leftrightarrow m \oplus \text{DES}_k^{-1}(c) &= m' \oplus \text{DES}_k^{-1}(c') \end{aligned}$$

- Nun kann der Angriff mit (m, c) -Paaren gestartet werden

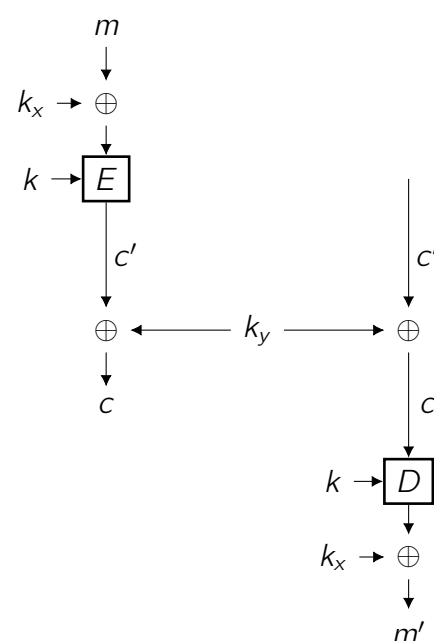


Abbildung 3: Konstruktion für den Angriff

Angriff:

Gegeben sind eine Menge von (m, c) -Paaren. Für alle (2^{56}) Schlüssel k :

1. Berechne $m \oplus \text{DES}_k^{-1}(c)$ für alle (m, c) -Paare
2. Falls für zwei unterschiedliche Paare (m, c) und (m', c) gilt, dass $m \oplus \text{DES}_k^{-1}(c) = m' \oplus \text{DES}_k^{-1}(c')$, dann sind die Paare ein "slid pair"-Kandidat
3. Berechne $k_y = c \oplus c'$
4. Berechne damit $k_x = m \oplus \text{DES}_k^{-1}(c \oplus k_y)$
5. Teste (k, k_x, k_y) mit weiteren Paaren auf Korrektheit

2.5.5 Aufwand für Advanced Slide-Attack gegen DES-X

- Mindestens $2^{64/2} = 2^{32}$ Klartext-Chiffre-Paare benötigt (≈ 32 GB)
- Für alle 2^{56} Schlüssel: $\approx 2^{32}$ DES-Operationen und Sortieren: $\approx 32 \cdot 2^{32}$
- Kollision auf Korrektheit prüfen (konstant)

Insgesamt also

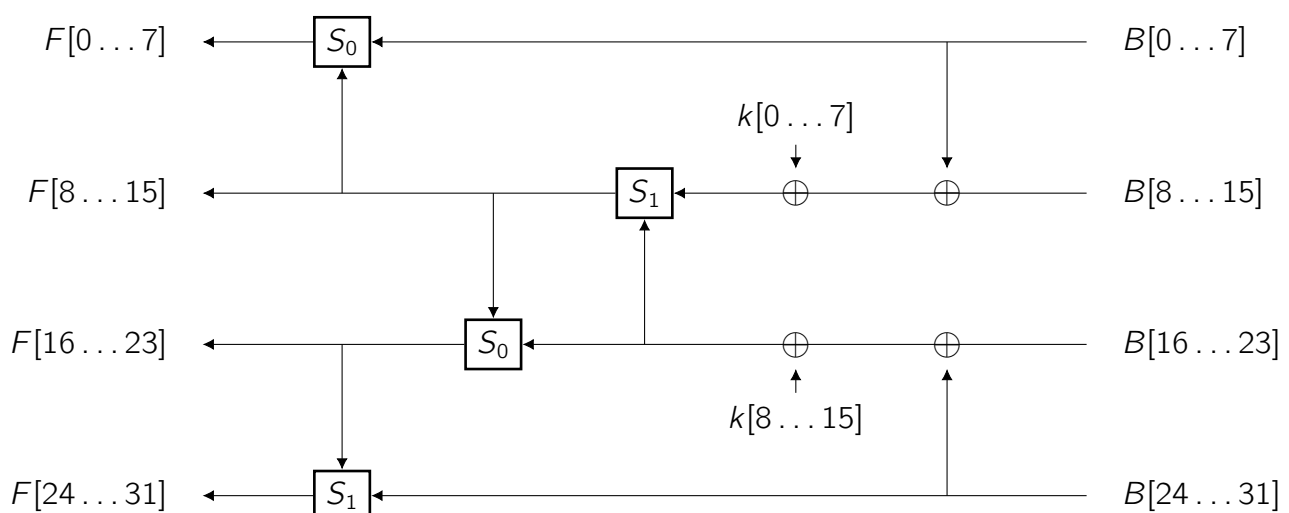
- $\approx 2^{88}$ ($= 2^{56} \cdot 2^{32}$) DES-Operationen
- $\approx 2^{32}$ benötigte (m, c) -Paare

3 Lineare Kryptoanalyse

Bei der **Linearen Kryptoanalyse** wird versucht, lineare Abhängigkeiten innerhalb eines Verschlüsselungssystems zu finden (auch wenn diese nur mit einer gewissen Wahrscheinlichkeit auftreten).

3.1 FEAL-4

3.1.1 F-Funktion

Abbildung 4: F -Funktion von FEAL-4

FEAL-4 verwendet wie DES eine Feistel-Struktur mit einer F -Funktion, die in Abbildung 4 dargestellt ist. Die S -Boxen führen folgende Berechnung durch:

$$S_i(a, b) := \text{rot2}((a + b + i) \bmod 256)$$

wobei *rot2* die Bits um zwei nach links verschiebt (also z.B. $\text{rot2}(00000001) = 00000100$).

3.1.2 Beobachtung

Die S-Boxen verhalten sich an einer Stelle linear: Das niederwertigste Bit von $(a + b) \bmod 256$ entspricht der Summe (dem XOR) der niederwertigsten Bits von a und b , da kein Übertrag auftreten kann. Unter Beachtung der Rotation gilt dann

$$\begin{aligned} S_0(a, b)[2] &= a[0] \oplus b[0] \\ S_1(a, b)[2] &= a[0] \oplus b[0] \oplus 1 \end{aligned}$$

Damit können wir folgende Gleichungen für das Ergebnis der F -Funktion aufstellen:

$$\begin{aligned} F[2] &= B[0] \oplus F[8] \\ F[10] &= 1 \oplus (B[8] \oplus B[0] \oplus k[0]) \oplus (B[16] \oplus B[24] \oplus k[8]) \end{aligned}$$

3.1.3 Charakteristik

FEAL-4 besteht aus 4 Runden, in denen die F -Funktion mit einem 16-Bit-Schlüssel ausgeführt wird (siehe Abbildung 5). Mithilfe der gerade aufgestellten Gleichungen lassen sich folgende Charakteristiken ableiten:

$$\begin{aligned} F_0[2, 8] &:= F_0[2] \oplus F_0[8] \\ &= B_0[0] \end{aligned}$$

$$\begin{aligned} R_1[2, 8] &= F_0[2, 8] \oplus m_l[2, 8] \\ &= B_0[0] \oplus m_l[2, 8] \\ &= m_l[0, 2, 8] \oplus m_r[0] \end{aligned}$$

$$\begin{aligned} R_1[2, 8] &= F_2[2, 8] \oplus R_3[2, 8] \\ &= B_2[0] \oplus R_3[2, 8] \\ &= L_3[0] \oplus R_3[2, 8] \end{aligned}$$

Durch Gleichsetzen erhalten wir:

$$m_l[0, 2, 8] \oplus m_r[0] = L_3[0] \oplus R_3[2, 8]$$

Dadurch haben wir eine Beziehung zwischen der Nachricht und dem Ergebnis der vorletzten Runde (L_3 und R_3) gefunden.

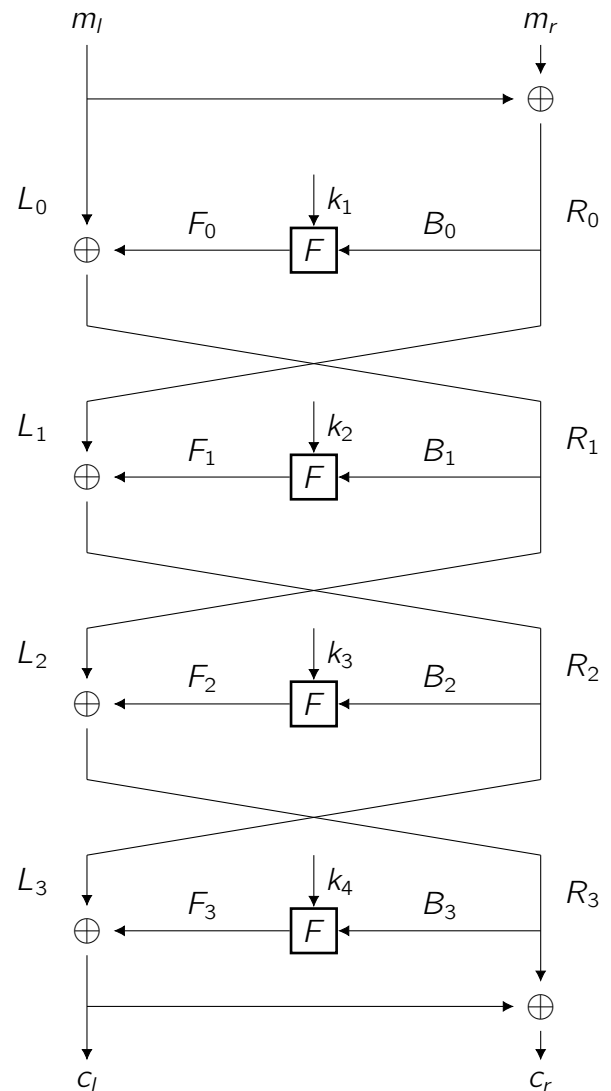


Abbildung 5: Struktur von FEAL-4 (vereinfacht)

3.1.4 Angriff

Gegeben sind eine Menge von (m, c) -Paaren. Für alle (2^{16}) Schlüssel k_4 :

1. Entschlüssele c mit k_4 , dadurch erhalten wir L_3 und R_3
2. Falls die gefundene Gleichung $m_l[0, 2, 8] \oplus m_r[0] = L_3[0] \oplus R_3[2, 8]$ nicht für alle (m, c) -Paare gilt, ist k_4 falsch
3. Wiederhole mit verschiedenen Charakteristiken, bis nur noch wenige k_4 übrig sind
4. Führe den Angriff analog gegen k_1 durch
5. Wenn k_1 und k_4 bekannt, können k_2 und k_3 durch vollständige Suche (jeweils 2^{16}) gefunden werden

3.1.5 Aufwand

- ≈ 16 (besser ≈ 20) Klartext-Chiffre-Paare benötigt
- Rechenaufwand $\approx 2^{16}$

3.2 DES

3.2.1 Angriff

Auch bei DES ist die lineare Analyse möglich, obwohl die S-Boxen (6 Eingänge, 4 Ausgänge) keine linearen Funktionen sind. Bei S_5 hat eine lineare Beziehung der Ein- und Ausgänge die Wahrscheinlichkeit $\frac{3}{16} \approx 0.19$.

Diese Beziehung lässt sich auf 3 Runden erweitern. Dann kann der Angriff wie bei FEAL-4 durchgeführt werden. Für 16 Runden (wie bei DES üblich) kann auch eine Beziehung gefunden werden, die Erfolgswahrscheinlichkeit sinkt dabei allerdings.

Details zum Angriff im Skript.

3.2.2 Aufwand

- $\approx 2^{47}$ Klartext-Chiffre-Paare benötigt
- "Zurückrechnen" der ersten und letzten Runde nur $2 \cdot 2^6$ (liefert 14 Bit des Schlüssels)
- Restliche 42 Bit des Schlüssels müssen durch vollständige Suche gefunden werden
- Schneller als Brute-Force (2^{56}), aber sehr viele Klartext-Chiffre-Paare benötigt

4 Differentielle Kryptoanalyse

Bei der **Differentiellen Kryptoanalyse** werden Differenzen zwischen Klartexten betrachtet (meistens per XOR), welche mit einer gewissen Wahrscheinlichkeit zu bestimmten Differenzen in den zugehörigen Chiffren führen.

4.1 DES

4.1.1 Idee

Wir betrachten nun nicht mehr (m, c) -Paare, sondern zwei Paare $(m_1, c_1), (m_2, c_2)$ und deren Differenz (m', c') mit $m' := m_1 \oplus m_2$ und $c' := c_1 \oplus c_2$.

Da der Schlüssel nur per XOR eingeht und sich dadurch die Differenzen der Zwischenergebnisse nicht ändern, können dadurch Informationen über den Schlüssel gewonnen werden.

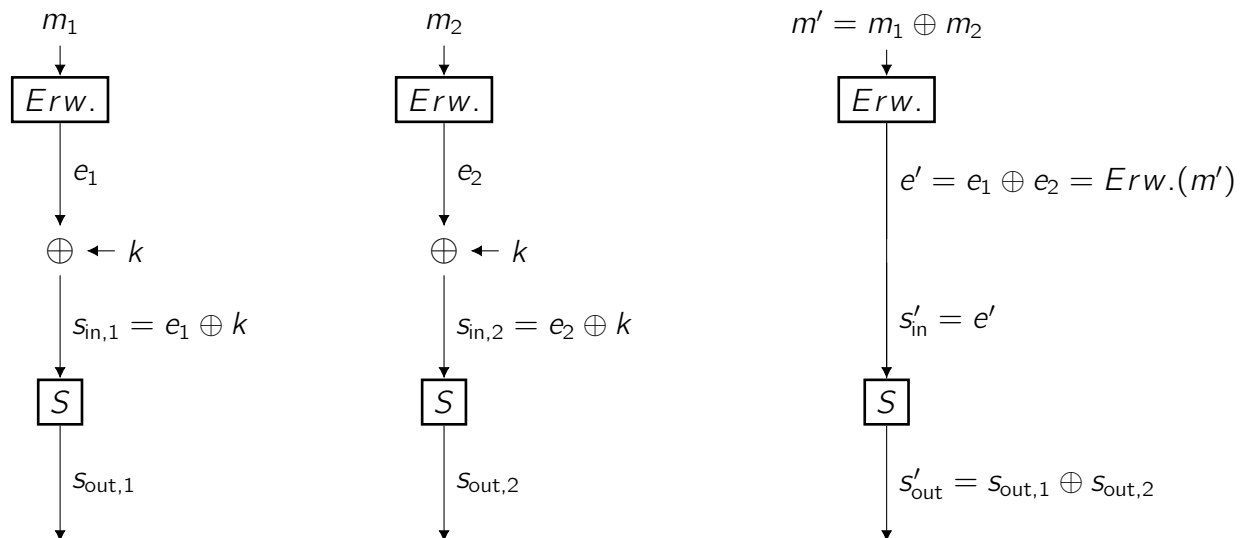


Abbildung 6: DES-S-Boxen und Zwischenergebnisse mit Eingaben und Differenzen

4.1.2 Angriff auf 1-Runden-DES

Beobachtung: $s'_{in} = e' = e_1 \oplus e_2$ ist unabhängig von k .

Wir erstellen uns nun für jede S -Box eine **Differenz-Tabelle**, die alle möglichen Kombinationen von Ein- und Ausgangsdifferenzen enthält (64×16 -Tabelle). Zeile i entspricht der Eingangsdifferenz $s'_{in} = i$. In Spalte j sind alle Paare $s_{in,1}$ und $s_{in,2}$ eingetragen, für die die Ausgangsdifferenz $s'_{out} = j$ ist.

1. Berechne s'_{in} und s'_{out} aus (m_1, c_1) und (m_2, c_2)
2. Lese mögliche $s_{in,1}$ und $s_{in,2}$ aus der Tabelle ab
3. Berechne alle Schlüsselkandidaten k durch $k = s_{in,1} \oplus e_1$ (gilt, da $s_{in,1} = e_1 \oplus k$)
4. Wiederhole mit anderen Paaren, bis nur noch ein Kandidat k übrig ist