Data Encryption Standard - DES

Simon Sternsdorf
Alexej Rotar

25. März 2016

Inhaltsverzeichnis

1 Einleitung

2 Mathematische Grundlagen

3 Kryptologische Grundlagen

3.1 Perfekte Sicherheit

3.2 Diffusion und Konfusion

Ein Verschlüsselungsverfahren, das die Redundanz, bzw. Struktur einer Nachricht beim Verschlüsseln beibehält, kann gut analysiert werden. Denn es kann dann der Geheimtext nach solchen Strukturen untersucht werden und so können gewisse Schlüssel mit höherer Wahrscheinlichkeit ausgewählt, bzw. ausgeschlossen werden. Beispielsweise kann eine Verschiebchiffre, wie die Caesar-Chiffre, untersucht werden, indem jedes Auftreten eines Buchstaben gezählt wird. Bei genügend langem, deutschen Text kann der häufigste Buchstabe in der Chiffre mit großer Wahrscheinlichkeit auf das „e“ abgebildet werden. So kann sofort der Schlüssel berechnet werden.

Ähnliche, statistische Analysen können auch auf andere Verfahren, die die Struktur erhalten, angewandt werden. Verfahren, die jedes Eingabezeichen auf genau ein Ausgabezeichen abbilden, erhalten im Allgemeinen die Redundanz. Von Shannon wurden zwei allgemeine Methoden entwickelt, die solche Analysen erschweren.

Diffusion Bei der Diffusion geht es darum, die Struktur einer Nachricht zu zerstören, sodass die Redundanz auf größere Strukturen verteilt wird. So tritt immer noch Redundanz auf, jedoch nur bei den größeren Strukturen. Die Wahrscheinlichkeiten für einzelne Strukturen sinken dadurch. Daher müssen wesentlich mehr Nachrichten abgefangen werden, um statistische Analysen betreiben zu können. Konkret bedeutet das, dass sich jedes Zeichen eines Klartexts auf möglichst viele Zeichen des Geheimtexts auswirken muss. Die Veränderung eines einzelnen Zeichens führt im Idealfall zur Veränderung eines jeden Zeichens im Geheimtext mit Wahrscheinlichkeit $1/2$.

Konfusion Konfusion veschleiert den Zusammenhang zwischen dem Geheimtext und dem Schlüssel. So wird ein statistischer Angriff ebenfalls erschwert. Denn selbst wenn gewisse Statistiken den Schlüsselraum einengen können, kann so trotzdem nur schwer der endgültige Schlüssel ermittelt werden, da nicht sofort ersichtlich ist, welche Schlüssel alle Statistiken erfüllen. Damit ein Verfahren konfus ist, sollte sich möglichst jeder Teil des Schlüssels auf die Geheimnachricht auswirken.

3.3 Nicht-Linearität

Ein weiteres Kriterium für die Sicherheit eines Verfahrens ist die *Nicht-Linearität*. Das bedeutet, dass die Verschlüsselung möglichst weit entfernt ist von einer linearen Abbildung. Der Grund dafür ist, dass sich solche Abbildungen gut

analysieren lassen. Denn bekanntermaßen lässt sich eine lineare Abbildung f schreiben als

$$f : x \mapsto A_f * x$$

mit A_f als Darstellungsmatrix von f. Damit f invertierbar ist, muss auch die Matrix A_f invertierbar und damit quadratisch sein. Um die n-dimensionale Matrix A_f zu ermitteln, benötigt man n linear unabhängige Vektoren p_i und deren Bilder c_i unter f. Dann kann man die einzelnen Gleichungen

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{pmatrix} * \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$$

zusammenfassen zu der Matrixgleichung

$$A_f * P = C$$

wobei $P = (p_1, \dots, p_n)$ und $C = (c_1, \dots, c_n)$. Da die Vektoren p_i linear unabhängig sind, lässt sich die Matrix invertieren und es gilt

$$A_f = C * P^{-1}$$

Ist die Matrix A_f bekannt, ermittelt man leicht f^{-1} indem man A_f^{-1} berechnet.

Konstruktion einer sicheren Chiffre Damit ein Verfahren konfus und diffus ist, muss es Nachrichten so verschlüsseln, dass die Wahrscheinlichkeiten für die unterschiedlichen Nachrichten gleich verteilt sind. Eine Abbildung, welche dieses durch mehrfache Anwendung bewerkstelligt, nennt man eine *mischende Abbildung*. Eine Möglichkeit, solche Abbildungen zu konstruieren, ist die abwechselnde Anwendung zweier nicht-kommutierender Abbildungen. Shannon hat vorgeschlagen, hierfür Permutationen und Substitutionen zu verwenden. Um eine Abbildung mehrfach anwenden zu können, muss sie *endomorph* sein. Das heißt, sie muss denselben Bild- und Urbildraum haben.

4 Blockchiffren

4.1 Allgemein

Im letzten Abschnitt wurde die Caesar-Chiffre erwähnt. Bei dieser handelt es sich um eine sogenannte *Stromchiffre*. Bei solchen Algorithmen wird jedes Zeichen einzeln verschlüsselt. Bei manchen Verfahren werden dagegen ganze Blöcke verschlüsselt. Dann spricht man von *Blockchiffren*. Dabei wird eine Nachricht M in mehrere Blöcke der Länge l aufgeteilt. Sollte l kein Teiler der Länge von M sein, wird der letzte Block mit weiteren Zeichen entsprechend aufgefüllt. Diese Zeichen nennt man auch *Padding*.
Nun wird jeder Block mit demselben Schlüssel k verschlüsselt. Man erhält dieselbe Anzahl verschlüsselter Blöcke. Zusammen bilden die Blöcke die verschlüsselte Nachricht C.

4.2 Betriebsmodi

Da eine Blockchiffre nur die Verschlüsselung eines Teils einer Nachricht festlegt, muss noch festgelegt werden, wie eine Verschlüsselung des gesamten Texts abläuft. Dafür gibt es unterschiedliche Möglichkeiten. Diese nennt man *Betriebsmodi*.

Electronic Codebook Beim ECB wird jeder Block unabhängig betrachtet und verschlüsselt. Die verschlüsselte Nachricht wird dann beschrieben durch

$$C = (c_1, \ldots, c_r) := (f(x_1, k), \ldots, f(x_r, k))$$

Da bei diesem Verfahren jeder Block unabhängig durch einen anderen ersetzt wird, könnte man es auch als Substitutionschiffre betrachten. Wie bei Substitutionschiffren üblich, wird somit zumindest die grobe Struktur der Nachricht erhalten. Daher lässt es sich, wie zuvor bereits geschildert, gut analysieren oder sogar unbemerkt manipulieren.

Cipher Block Chaining Um die eben erwähnten Manipulationsmöglichkeiten zu verhindern, wird beim CBC zur Verschlüsselung eines Blocks der verschlüsselte vorige Block mitverwendet:

$$c_i := f(x_i \oplus c_{i-1}, k)$$

Dabei wird für c_0 ein beliebiger Initialisierungsblock verwendet. Bei Anwendung der Entschlüsselungsfunktion g erhält man

$$\begin{aligned} g(c_i) &= g(f(x_i \oplus c_{i-1}, k)) \\ &= x_i \oplus c_{i-1} \end{aligned}$$

Durch Addition mit c_{i-1} erhalten wir den Klartextblock. Das heißt

$$m_i = c_{i-1} \oplus g(c_i, k)$$

Da sich also der Vorgängerblock ebenfalls auf die Entschlüsselung eines Blocks auswirkt, würde die Veränderung eines Geheimtextblocks eine Zerstörung des nachfolgenden Blocks bewirken, wodurch eine Manipulation erkannt werden könnte.

Output Feedback Mode Im OFB werden zu einem beliebigen Initialisierungsblock I_1 rekursiv weitere Blöcke mittels der Blockchiffre berechnet und auf den Klartextblock addiert. Die Vorschrift sieht formal folgendermaßen aus:

$$\begin{aligned} I_{i+1} &:= f(I_i, k) \\ c_i &:= x_i \oplus I_{i+1} \end{aligned}$$

Gewissermaßen könnte man die Blockchiffre in diesem Modus als Zufallszahlengenerator betrachten. Es ist hier tatsächlich nicht notwendig, dass die Blocklänge von I_i und x_i gleich sind. Die Länge von I_i muss mindestens so groß

sein, wie die von x_i. Sollte sie größer sein, werden die überflüssigen Stellen einfach abgeschnitten.
Das Problem an diesem Modus ist, dass die Blöcke wieder unabhängig von einander betrachtet werden und sich somit dieselben Schwierigkeiten wie beim ECB ergeben.

Cipher Feedback Mode Auch beim CFB wird die Blockchiffre als Zufallszahlengenerator betrachtet. Jedoch werden hier wie auch beim CBC bereits berechnete Blöcke für die Berechnung der weiteren Blöcke verwendet:

$$I_{i+1} := f(c_i, k)$$

Auch hier können die Längen verschieden sein und müssen durch entsprechendes Abschneiden von Zeichen kompensiert werden.

4.3 Feistel-Chiffren

Eine Spezialisierung der Blockchiffren sind die sogenannten *Feistel-Chiffren*. Im Folgenden soll zunächst auf ihre Struktur und anschließend auf ihre Entschlüsselung eingegangen werden.

Struktur Eine Feistel-Chiffre ist eine endomorphe Blockchiffre mit einer festgelegten Rundenanzahl. Sie wird festgelegt durch eine sogenannte *innere Blockchiffre* B, eine Rundenanzahl $N \in \mathbb{N}$, eine Schlüsselmenge $\tilde{K}$ und eine Schlüsselauswahlfunktion ϕ. Dabei ist B definiert durch

$$B := (\mathbb{Z}_2^n, \mathbb{Z}_2^n, K, f)$$

wobei K eine weitere Schlüsselmenge und $f : \mathbb{Z}_2^n \times K \to \mathbb{Z}_2^n$ die Verschlüsselungsfunktion sind. Dann ist die Feistel-Chiffre definiert durch

$$F := (\mathbb{Z}_2^{2n}, \mathbb{Z}_2^{2n}, \tilde{K}, \tilde{f})$$

Nun ist $\tilde{f} : \mathbb{Z}_2^{2n} \times \tilde{K} \to \mathbb{Z}_2^{2n}$ wie folgt definiert.

Zu Beginn wird eine Klartextnachricht $P \in \mathbb{Z}_2^{2n}$ in eine linke und eine rechte Hälfte $L_0,\ R_0 \in \mathbb{Z}_2^n$ geteilt. Das heißt $P = (L_0, R_0)$ Anschließend wird rekursiv berechnet

$$L_i := R_{i-1} \tag{1}$$
$$R_i := L_{i-1} \oplus f(R_{i-1}, k_i) \tag{2}$$

für $i \in \{1, \ldots, N\}$. Dabei ist $\oplus$ eine bitweise XOR-Verknüpfung. Der Vektor $(k_1, \ldots, k_N)$ wird durch die Funktion $\phi : \tilde{K} \to K^n$ vorgegeben. Die k_i werden als *Rundenschlüssel* bezeichnet. Die Funktion

$$\tilde{f} : \mathbb{Z}_2^{2n} \times \tilde{K} \to \mathbb{Z}_2^{2n}, (P, k) \mapsto (R_N, L_N)$$

beschreibt schließlich eine Verschlüsselung mit einer Feistel-chiffre. Hier ist insbesondere zu beachten, dass nun R_N links, während L_N rechts steht. Der Grund hierfür wird im nächsten Abschnitt ersichtlich. Der Ablauf kann folgendermaßen veranschaulicht werden:

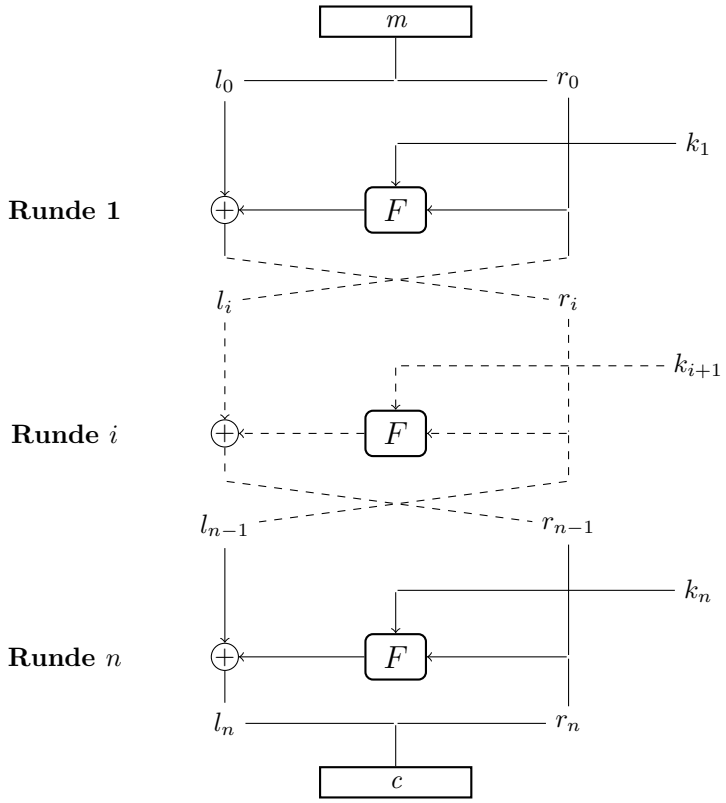

Abbildung 1: Ablauf einer Feistel-Chiffre

Entschlüsselung Es sei die Chiffre $C := (R_N, L_N)$ unser Ausgangspunkt. Nun soll hieraus wieder die ursprüngliche Nachricht $P := (L_0, R_0)$ ermittelt werden. Wir wollen zeigen, dass sich die Nachricht mittels Anwendung desselben Algorithmus bestimmen lässt. Dazu zeigen wir zunächst ein Lemma.

Lemma 4.1. *Der Algorithmus lässt sich umkehren unabhängig davon, wie die Funktion f definiert ist.*

Proof. Durch Umformung von 1, bzw. 2 nach R_{i-1}, bzw. L_{i-1} erhält man

$$R_{i-1} = L_i \tag{3}$$

$$\begin{aligned} L_{i-1} &= R_i \oplus f(R_{i-1}, k_i) \\ &= R_i \oplus f(L_i, k_i) \end{aligned} \tag{4}$$

□

Es kann also aus einem Nachfolger der entsprechende Vorgänger ermittel werden.

Satz 4.1. *Die Umkehrung des Algorithmus ist gerade derselbe Algorithmus, wobei die Rundenschlüssel in umgekehrter Reihenfolge verwendet werden.*

Proof. Es wurde bereits festgestellt, dass in der resultierenden Chiffre die linke und rechte Hälfte aus dem letzten Schritt vertauscht sind. Wir definieren daher die beiden Hälften neu:

$$L'_0 := R_N$$
$$R'_0 := L_N$$

Wir definieren weiterhin $j := N-i$, also folgt $i = N-j$. Es folgt $N-(j+1) = N-j-1 = i-1$. Da nach jeder Runde die linke und rechte Hälfte vertauscht werden, ergibt sich mit dieser Definition

$$R'_j := L_{N-j} = L_i$$
$$L'_j := R_i$$

und außerdem

$$R'_{j+1} = L_{i-1}$$
$$L'_{j+1} = R_{i-1}$$

Durch Einsetzen in 3, bzw. 4 ergibt sich

$$L'_{j+1} = R'_j$$
$$R'_{j+1} = L'_j \oplus f(R'_j, k_i)$$

Nach einem Indexshift und Umbenennung der Rundenschlüssel zu $k'_j := k_{N-(j-1)} = k_{N-j+1} = k_{i+1}$ ergibt sich gerade die ursprüngliche Rekursionsformel

$$L'_j = R'_{j-1}$$
$$R'_j = L'_{j-1} \oplus f(R'_{j-1}, k'_j)$$

□

Die Umbenennung der Schlüssel bedeutet dabei, dass die Schlüssel in umgekehrter Reihenfolge angewandt werden. Der Versatz um eins ist notwendig, da wir bei den Schlüsseln bei 1 zu zählen beginnen.

5 DES

5.1 Allgemein

Der DES-Algorithmus ist eine Feistel-Chiffre mit interner Blockchiffre

$$B = (\mathbb{Z}_2^{32}, \mathbb{Z}_2^{32}, \mathbb{Z}_2^{48}, f)$$

und $N = 16$ Runden. Nach der Definition der Feistel-Chiffren folgt daraus, dass 64-Bit-Blöcke verschlüsselt werden und die Rundenschlüssel 48-Bit breit sind. Auf die Funktion f wird anschließend genauer eingegangen.

Eine Besonderheit des DES sind die Eingangs- und Ausgangspermutation IP, bzw. FP (initial, bzw. final permutation). Dabei ist die Ausgangspermutation definiert als

$$FP := IP^{-1}$$

Sie werden einmalig vor, bzw. nach dem Algorithmus ausgeführt. Da die Permutationen bekannt sind und nur am Anfang, bzw. am Ende ausgeführt werden, tragen sie nicht direkt zur Sicherheit des Verfahrens bei. Ein möglicher Grund für deren Verwendung könnte deren einfache hardwaretechnische Implementierung sein. Denn um eine Permutation in Hardware zu realisieren, kann man einfach entsprechend Ein- und Ausgänge direkt verdrahten. Softwaretechnische Implementierungen sind dagegen wesentlich aufwändiger und auch weniger effizient. Dadurch verlieren auch mögliche softwarebasierte Angriffe an Effizienz. Bei den heutigen Prozessoren spielt das allerdings keine Rolle mehr.

Aufgrund der Definition von FP als Umkehrung von IP lässt sich auch der DES-Algorithmus entschlüsseln, indem derselbe Algorithmus erneut angewandt wird, so wie es bei den Feistel-Chiffren üblich ist.

5.2 Die Rundenfunktion

Die Funktion f ist der Kern des DES-Verfahrens. Sie ist definiert als

$$f(R, k) \mapsto \pi(S(E(R) \oplus k))$$

wobei k der Rundenschlüssel ist und $R \in \mathbb{Z}_2^{32}$ die aktuell rechte Hälfte. Die einzelnen Funktionen sind folgendermaßen definiert:

Die Expansion Die Abbildung $E : \mathbb{Z}_2^{32} \to \mathbb{Z}_2^{48}$ ist definiert durch die Tabelle 1.

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Tabelle 1: Expansionsabbildung

Es fällt auf, dass stets das letzte und das erste Bit eines 4-Bit-Blocks verdoppelt werden. Das heißt es werden 16 Bits ergänzt, sodass der resultierende Block 48 Bit breit ist.
Da die Expansion leicht als Matrixmultiplikation dargestellt werden kann, ist E eine lineare Abbildung. Auch dient sie weder der Diffusion noch Konfusion,

denn sie verstreut die Nachricht nicht, sondern behält zum größten Teil ihre Struktur bei. Allerdings ist sie notwendig, um die Nachricht auf die Größe eines Rundenschlüssels zu bringen. Dieser wird anschließend auf das Ergebnis addiert.

Die Substitution Die Abbildung $S : \mathbb{Z}_2^{48} \to \mathbb{Z}_2^{32}$ bildet den nun entstandenen 48-Bit-Block auf die urpsprüngliche Größe von 32 Bit ab, damit mit diesem anschließend, wie von Feistel-Chiffren gewohnt, weitergearbeitet werden kann. Dabei werden insgesamt acht 6-Bit-Blöcke durch wiederum acht 4-Bit-Blöcke ersetzt. Dazu werden die sogenannten *S-Boxen* verwendet. Es dient stets der erste Block als Eingabe für S_1, der zweite für S_2 und so weiter. Die folgende Tabelle zeigt exemplarisch eine solche S-Box.

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

Tabelle 2: S-Box S_5

Aus der Abbildung erkennt man, dass eine S-Box vier Zeilen enthält. Jede dieser Zeilen ist eine Substitution. Welche Substitution verwendet wird, entscheiden das Bit 1 und 6 des Eingabeblocks. Das heißt, es wird für einen Block $(b_1, b_2, b_3, b_4, b_5, b_6)$ die Substitution $b_1 b_6$ verwendet. Die übrigen Stellen, die Zahlen von 0 bis 15 darstellen können, bilden die zu ersetzende Zahl, entscheiden also, welche Spalte einer S-Box verwendet wird.

Als Beispiel sei die Bitfolge 011101 der sechste Block und damit die Eingabe für oben abgebildete S-Box S_5. Als Zeile erhalten wir dann $01_2 = 1_{10}$ und als Spalte $1110_2 = 14_{10}$. Somit erhalten wir $8_{10} = 1000_2$ als Ausgabe.
Man sieht leicht, dass die Substitution keine lineare Abbildung ist.

Proof. Um die nicht-Linearität zu zeigen, müssen wir zeigen, dass

$$\exists b_1, b_2 : S(b_1 \oplus b_2) \neq S(b_1) \oplus S(b_2)$$

gilt. Wir definieren $b_1 := 000000$, bzw. $b_2 := 000001$ und betrachten diese wieder als Eingabe von S_5.

$$\begin{aligned} S(000000 \oplus 000001) &= S(000001) \\ &= 1110_2 \\ S(000000) \oplus S(000001) &= 0010_2 \oplus 1110_2 \\ &= 1100_2 \\ &\neq 1110_2 \end{aligned}$$

□

Tatsächlich könnte es ein Kriterium für die Wahl genau dieser S-Boxen sein, eine möglichst nicht-lineare Abbildung S zu bilden.

Die Permutation Die Abbildung $\pi : \mathbb{Z}_2^{32} \to \mathbb{Z}_2^{32}$ permutiert schließlich das Ergebnis der Substitution gemäß Tabelle 3.

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Tabelle 3: Permutationsabbildung

Wie auch bei der Expansion handelt es sich hier um eine lineare Abbildung. Im Gegensatz zur Expansion zerstört sie allerdings die Struktur des Blocks. Dadurch wird bewirkt, dass einzelne Bits als Eingabe für möglichst viele S-Boxen dienen und sich damit möglichst auf die gesamte Chiffre auswirken. Folglich dient sie der Diffusion.

5.3 Die Schlüsselauswahlfunktion

Wie aus dem Abschnitt über Feistel-Chiffren bekannt, ist eine Schlüsselauswahlfunktion ϕ notwendig, die aus der Schlüsselmenge $\mathbb{Z}_2^{64}$ die 16 Rundenschlüssel bildet. Beim DES werden nur 56-Bit des Schlüssels berücksichtigt, da die übrigen 8 Paritätsbits sind. Daher werden zunächst die 56 relevanten Bits ausgewählt und auf zwei 28-Bit Register, C und D, verteilt. Dazu wird die Permutation PC-1 verwendet. Anschließend werden die 16 Rundenschlüssel gebildet, indem zunächst beide Register nach folgendem Schema rotiert werden:

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

Das heißt, es wird jeweils vor der entsprechenden Runde rotiert. Nach jeder Rotation wird mittels der Permutation PC-2 der entsprechende Rundenschlüssel gebildet.

Ähnlich wie die Anfangs- und Endpermutationen wirkt sich PC-1 nicht auf die Sicherheit des Verfahrens aus, da sie nur einmalig zu Beginn angewandt wird. Sie dient im Grunde lediglich der Auswahl der für den Schlüssel relevanten Bits. Die Rotation und PC-2, dagegen, sorgen dafür, dass die Bits des Schlüssels sich an möglichst vielen Stellen auf die Chiffre auswirken und damit für eine hohe Konfusion. Denn es fällt auf, dass die Summe der 1-Bit-Rotationen 28 ergibt, also gerade die Größe eines Registers. Damit wird gewährleistet, dass die Schlüsselbits sich gleichmäßig über die gesamte Chiffre verteilen. Der letzte Schlüssel k_{16} wird aus den ursprünglichen Registern C und D gebildet.

5.4 Sicherheit

Bei der Betrachtung der einzelnen Teilfunktionen des DES wurden bereits deren Auswirkungen auf die Sicherheit angesprochen. Im Großen und Ganzen sieht man nun, dass das DES sowohl einen hohen Grad an Diffusion wie an Konfusion aufweist. Es fällt außerdem auf, dass hier genau die von Shannon vorgeschlagene Vorgehensweise verwendet wird. In jeder Runde findet eine Permutation

und eine Substitution statt und das Verfahren wird iteriert. Die obigen Untersuchungen bezüglich der Beiträge zur Sicherheit können somit auch als Veranschaulichung an einem konkreten Beispiel für Shannons abstrakte Idee dienen.

Als einzige nicht-lineare Funktion sorgt die Substitution auf für die nicht-Linearität des gesamten Verfahrens und ist damit ein entscheidender Faktor für dessen Sicherheit. Damit ist die richtige Wahl der S-Boxen von großer Bedeutung. Es hat sich herausgestellt, dass bei kleinen Änderungen einer der S-Boxen das Verfahren bereits wesentlich unsicherer wird. Wie genau die richtigen S-Boxen ermittelt wurden ist allerdings nicht bekannt.

6 Kryptoanalyse

Bei der Kryptoanalyse geht es darum, aus abgefangenen Nachrichten Informationen über die Klartextnachricht oder den Schlüssel abzuleiten. Man unterscheidet dabei unterschiedliche Arten. Für uns sind die relevanten *Known-Plaintext-Angriff* und *Chosen-Plaintext-Angriff*. Bei ersterem kennt der Angreifer zu einem Geheimtext auch den Klartext, während er bei letzterem zu jedem beliebigen Klartext den Geheimtext erlangen kann.
Es wurden für einen Angriff gegen den DES zwei Methoden vorgestellt. In der *Linearen Kryptoanalyse* versucht man die Verschlüsselung durch eine lineare Abbildung anzunähern, da solche, wie bereits erwähnt, leichter zu analysieren sind. Wir beschränken uns auf die *Differentielle Kryptoanalyse*.

6.1 Das Ziel

Das Ziel der differentiellen Kryptoanalyse ist es, einen Teilschlüssel zu bestimmen. Hat man beispielsweise einen gesamten 48-Bit-Rundenschlüssel gefunden, so bleiben nur noch 8 relevante Bits übrig, die leicht mittels Brute-Force ermittelt werden können. Im Folgenden gelten die Bezeichnungen

- x, x' : Eingaben für eine Rundenfunktion
- y, y' : Ausgaben einer Rundenfunktion
- $\Delta x, \Delta y$: Ein-,Ausgabedifferenz der Rundenfunktion
- e, a : Ein-, Ausgabe einer S-Box
- $\Delta e, \Delta a$: Ein-, Ausgabedifferenz einer S-Box

Aufgrund der Definition einer Rundenfunktion gilt der Zusammenhang

$$k^{(1)} = e^{(1)} \oplus E^{(1)}(x)$$

Wobei damit jeweils die ersten 6 Bit des Schlüssels, der Eingabe, bzw. der Expansion gemeint sind. Man kann also leicht einen Teilschlüssel ermitteln, wenn e bekannt ist. Dieses kann allerdings nicht ohne Weiteres ermittelt werden, wenn der Schlüssel unbekannt ist. Daher betrachtet man bei der Differentiellen Kryptoanalyse Differenzen anstatt einzelner Werte.

6.2 Differenzenverteilung

Es seien nun x, x', y, y' gegeben und damit auch $\Delta x = x \oplus x'$, bzw. $\Delta y = y \oplus y'$. Aus Δy lässt sich leicht Δa bestimmen durch

$$\Delta a = (P^{-1}(y)) \oplus (P^{-1}(y') = P^{-1}(\Delta y)$$

Wobei die letzte Gleichheit gilt, weil P eine lineare Abbildung ist. Für die Eingabedifferenz gilt

$$\Delta e = (E(x) \oplus k) \oplus (E(x') \oplus k) = E(\Delta x)$$

Es lässt sich also auch die Eingabedifferenz ohne Kenntnis des Schlüssels bestimmen. Nun stellt sich die Frage, inwiefern Ein- und Ausgabedifferenzen der S-Boxen zusammenhängen. Da die S-Boxen nicht-lineare Abbildungen sind, lässt sich kein Zusammenhang wie bei den Abbildungen E und P herstellen. Jedoch kann man bei bekanntem Δx 64 unterschiedliche Paare x, x' finden, die diese Differenz bilden. Nun kann zu jedem dieser Paare die Ausgabedifferenz

$$\Delta a = S(x) \oplus S(x')$$

betrachtet und mit der gegebenen verglichen werden. Im Allgemeinen werden mehrere Eingabepaare dieselbe Ausgabedifferenz erzeugen, da S keine Injektive Abbildung ist. Jedes dieser Paare ist potentiell richtig und bildet mit obiger Formel einen Schlüssel. Die Menge dieser potentiellen Schlüssel wird Kandidatenmenge $\mathcal{K}_{x,x'}$ genannt. Wiederholt man diesen Vorgang mit unterschiedlichen Eingabedifferenzen, ergeben sich unterschiedliche Kandidatenmengen, von denen aber jede den richtigen Schlüssel enthält. Die Schnittmenge einiger solcher Mengen ergibt schließlich den richtigen Schlüssel.
Die unterschiedlichen Ein- und Ausgabedifferenzen mit entsprechenden Eingabepaaren können in 64 Tabellen mit 64 Zeilen und 16 Spalten pro S-Box festgehalten werden. Die Anzahl der Eingabepaare, die eine bestimmte Ausgabedifferenz erzeugen, kann ebenfalls in einer Tabelle festgehalten werden, die *Differenzenverteilung* genannt wird.

6.3 Bestimmung der Differenzen

Betrachtet man nur eine einzelne Runde des DES, ist es leicht die Differenzen zu bestimmen

$$\begin{aligned}\Delta x_r &= \Delta M_r \\ \Delta y_r &= \Delta M_l \oplus \Delta C_l\end{aligned}$$

wobei ΔM und ΔC Klartext- und Geheimtextdifferenzen sind. Ähnliche Methoden existieren auch für bis zu fünf Runden. Ab sechs Runden ist das nicht mehr mit Sicherheit möglich. Stattdessen kann man nur noch mit gewissen Wahrscheinlichkeiten die Differenzen vorhersagen.

Definition 6.1. *Es sei $P(S_j, \Delta e \to \Delta a)$ die Wahrscheinlichkeit, dass die Eingabedifferenz Δe die Ausgabedifferenz Δa erzeugt. Es gilt*

$$P(S_j, \Delta e \to \Delta a) := \frac{|\{(e, e') | e \oplus e' = \Delta e \wedge S_j(e) \oplus S_j(e') = \Delta a\}|}{64}$$

Definition 6.2. *Es sei weiterhin $P(f, \Delta x, \to \Delta y)$ die Wahrscheinlichkeit, dass Δx mit F die Differenz Δy erzeugt. Es gilt*

$$P(F, \Delta x \to \Delta y) := \frac{|\{(x, x', k) | x \oplus x' = \Delta x \wedge F(k, x) \oplus F(k, x') = \Delta y\}|}{2^{32} \cdot 2^{48}}$$

Satz 6.1. *Zwischen den eben definierten Wahrscheinlichkeiten gilt nun folgender Zusammenhang:*

$$P(F, \Delta x \to \Delta y) = \prod_{j=1}^{8} P(S_j, \Delta e^{(j)} \to \Delta a^{(j)})$$

Nun wollen wir unsere Differenzen über mehrere Runden hinweg nachverfolgen. Dafür brauchen wir die sogenannte *DES-Charakteristik*.

Definition 6.3. *Die DES-Charakteristik ist definiert als $\Gamma := (\Delta m, \lambda, \Delta c)$, wobei*

- *Δm die Eingabedifferenz des DES,*
- *Δc die Ausgabedifferenz des DES und*
- *$\lambda := (\lambda_1, \ldots, \lambda_n)$ eine Liste mit Differenzen der einzelnen Runden ist. Dabei ist $\lambda_i = (\Delta x_i, \Delta y_i)$.*

Es erzeugt Δm die Ausgabe Δc gemäß Γ mit der Wahrscheinlichkeit

$$p^{\Gamma} := \prod_{i=1}^{n} P(F, \Delta x_i \to \Delta y_i)$$

Wir wollen nun Nachrichten unterscheiden, welche die richtige Ausgabedifferenz erzeugen, von solchen, die falsche erzeugen.

Definition 6.4. *Es sei ein Schlüssel K und eine Charakteristik Γ gegeben. Das Klartextpaar M, M' heißt (Γ, K)-treu, falls die folgenden Bedingungen gelten:*

- *$\lambda_i = (\Delta x_i, \Delta y_i)$ für alle i. Dabei sind Δx_i, bzw. Δy_i die Ein- und Ausgabedifferenzen der Rundenfunktionen bei der Verschlüsselung von M und M'.*
- *$C \oplus C' = \Delta c$.*

Nun ist ein Paar M, M' mit der Wahrscheinlichkeit p^{Γ} (Γ, k)-treu. Es kann jetzt wie zuvor beschrieben eine Kanditatenmenge für den Schlüssel aus einem Klartextpaar und einer Geheimtextdifferenz abgeleitet werden. Sollte das Klartextpaar (Γ, K)-treu sein, enthält die Kanditatenmenge auf jeden Fall den richtigen Schlüssel. Andernfalls ist das nicht zwangsläufig der Fall. Daraus folgt also, dass mit Wahrscheinlichkeit p^{Γ} der richtige Teilschlüssel in einer Kanditatenmenge enthalten ist. Es hat sich gezeigt, dass der richtige Teilschlüssel am häufigsten in den Mengen vorkommt. Daher kann dieser gefunden werden, wenn eine genügend hohe Anzahl Klartextpaare überprüft werden.

7 Schluss