DER DES ALGORITHMUS

ALEXEJ ROTAR & SIMON STERNSDORF[1]

30.3.2016

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

1 EINLEITUNG – SIMON STERNSDORF

Verschlüsselung fand schon im alten Rom Anwendung. Einer der ersten und wohl auch bekanntesten Verschlüsselungsalgorithmen der Werlt, die sogeannte “Cäsar-Chiffre“, findet man heutzutage in Popkultur um die ganze Welt. Es ist ein sehr einfaches symmetrisches Verfahren, bei dem das uns bekannte Alphabet zum Einsatz kommmt. Jeder Buchstabe wird durch einen anderen Buchstaben aus unserem Alphabet ersetzt, der um eine bestimmte Anzahl weiter hinten in der Reihenfolge steht. Das Ganze ist zyklisch und ergibt am Ende eine einfache Umwandlungstabelle. [3] Dieser Algorithmus ist heutzutage natürlich viel zu leicht zu knacken mit gerade einmal 25 verschiedenen Verschlüsselungsmöglichkeiten. Aber das Prinzip des symmetrischen Verschlüsselns mittels einer Chiffre ist uns bis heute erhalten geblieben. Der hier behandelte Algorithmus arbeitet im Grunde ganz ähnlich: Der DES Algorithmus.

1.1 Geschichte

Der DES Algorithmus ist trotz seiner Sicherheitsprobleme immer noch einer der weltweit am weitesten verbreiteten Verschlüsselungsalgorithmen. Nicht umsonst heißt er Data Encryption Standart. Vor allem in dem Derivat “triple-DES“ oder auch verkürzt “3DES“ wird er noch eingesetzt und wird es wohl auch noch viele Jahre lang werden. Er wurde 1974 in Folge einer Ausschreibung des NBS, des National Bureau of Standars, für einen sicheren Verschlüsselungsstandart zum Verschicken von Daten in den USA von IBM eingereicht. IBM beschäftigte damals unter anderem Horst Feistel, auf dessen Chiffren später genauer eingegangen wird. DES basiert grob auf dem schon zuvor entwickelten Algorithmus Luciver. [4] Die Verschlüsselung wurde 1976 nach Anpassungen der NSA, der National Security Agency, als allgemeiner Standart für verschlüsselte Datenverbindungen im Internet übernommen, und wurde vor allem von der Bankenindustrie und der US-Amerikanischen Regierung für die Kommunikation eingesetzt. [4]

1.2 Kritik an Sicherheit

Die Anpassungen der NSA sind sehr umstritten. So soll etwa der Sicherheitsgrad gesenkt worden sein durch Verkürzung der Schlüssellänge von 128 Bit auf 56 Bit und eventuell sogar die zur Verschlüsselung notwendigen Substitutions-Boxen von der NSA verändert worden sein.Man soll versucht haben sich eine Hintertür, einen sogenannten “backdoor“ in den Algorithmus einzubauen. Da DES von Anfang an nur für den normalen Datenverkehr, aber nicht für Dokumente der höchsten Sicherheitsstufe eingesetzt wurde, ist es bis heute sehr umstritten wie sicher DES wirklich ist. [5] Allerdings haben sich die meisten dieser Spekulationen nicht bestätigt. [1, Seite 44]

2 MATHEMATISCHE GRUNDLAGEN – SIMON STERNSDORF

2.1 Permutation und Expansion

Der DES-Algorithmus arbeitet mit sehr einfachen mathematischen Methoden. Der Schlüssel ist hierbei immer 64 Bit lang, wobei immer das letzte von 8 Bit als Korrektur-Bit verwendet wird, mit dem Speicher- und Übertragungsfehler ausgeglichen werden können. [6, Seite 104] Dieser Schlüssel wird so zum Verschlüsseln auf die Nachricht angewandt, die jeweils in 64 Bit Blöcke unterteilt wird. Eine Permutation auf diese Bits entspricht einem Verschieben der Bits auf eine neue Position innerhalb des Blocks. Das Ganze wird gerne als Tabelle dargestellt, als sogenannte Permutations-Tabelle.

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Abbildung 1: Eingangs-Permuations Tabelle[6, Seite 105]

Jede Zahl beziffert genau die neue Position des Bits innerhalb des Permutatins-Blocks. So wird hier das erste Bit auf das 58te abgebildet, das zweite auf das 50te und so weiter. Nach dem selben Prizip wird dann die Expansion angewandt, allerdings ist dabei die Tabelle größer als der ursprüngliche Block. Das heißt hier werden manche Bits auf mehrere Stellen gesetzt.

2.2 Substitution

Weiter wichtig sind zudem Substitutionen. Auf ihren Nutzen für die Sicherheit des Verfahrens wird später noch genauer eingegangen. Im Allgemeinen wird bei der Substitution ein Block an Bits durch einen anderen ersetzt. Dies geschieht wiederum durch Tabellen, wobei man hier beim DES auch von einer gleichzeitigen Kontraktion sprechen kann, da die Bitfolge verkürzt wird. [6, Seite 107]

Expansion und Permutation sind lineare Abbildungen, wobei die Permutation eine Abbildung $\mathbb{Z}_2^{32} \to \mathbb{Z}_2^{32}$ und die Expansion eine Abbildung E: $\mathbb{Z}_2^{32} \to \mathbb{Z}_2^{48}$sind. Die Substitution ist eine nicht-lineare Abbildung. [1, Seite 52]

3 KRYPTOLOGISCHE GRUNDLAGEN

3.1 Kryptosysteme - Simon Sternsdorf

Wir verwenden in dieser Arbeit die Definition eines Kryptosystems wie folgt: Ein Kryptosystem ist ein Tupel(P, C, K, f, g) wobei P, C und K nicht leer sein dürfen. Hierbei ist:

- **P** die Klartextmenge, sprich der Text der verschlüsselt werden soll
- **C** die Geheimtextmenge, sprich der verschlüsselte Text
- **K** die Schlüsselmenge, also der Schlüssel der zum verschlüsseln der Klartextmenge zur Geheimtextmenge verwendet wurde
- **f** die Verschlüsselungsfunktion, eine Abbildung $f : P \times K \to C$, die aus der Klartextmenge die Geheimtextmenge bildet
- **g** die Entschlüsselungsfunktion, eine Abbildung $g : C \times K \to P$, die aus der Geheimtextmenge die Klartextmenge bildet

Weiter muss gelten das $\forall k \in K : \exists k' \in K : g_{k'} \circ f_k = id_P$, was gleichbedeutent ist mit Injektivität und Surjektivität der Abbildung. Ohne diese Vorraussetzung wäre eine verschlüsselte Nachricht nicht mehr entschlüsselbar. Bei DES gilt $k' = k$. [1, Seite 9]

3.2 Perfekte Sicherheit - Simon Sternsdorf

3.2.1 *Konzept*

Um zu verstehen wie sicher oder unsicher der DES-Algorithmus ist, braucht man das Konzept der perfekten Sicherheit.
Hierbei gehen wir davon aus, dass der Angreifer über unendliche Rechenkapazitäten verfügt. Weiterhin haben wir ein Kryptosystem $\pi = (P, C, K, f, g)$, das ein nach Kerckhoff's Prinzip sicheres Verschlüsselungsverfahren enthält, was heißt f und g sind allgemein bekannt. [7] Wir haben nun Wahrscheinlichkeits-Verteilungen Ws auf P,C und K, zudem sind P, C und K die Zufallsvariablen für die Ws-Verteilungen. Nun ziehen wir ein $p \in P$ mit $Ws[P = p]$ und analog für C und K. Die Wahrscheinlichkeit für $P = p$ und $C = c$ sollen dabei > 0 sein. Nun ist unser Kryptosystem und damit unser Verschlüsselungsverfahren sicher wenn der folgende Satz gilt:

Satz 3.1 (Chiffretext-Verteilung)**.**

$$Ws[P = p|C = c] = Ws[P = p]|\forall p \in P, c \in C$$

Proof. Annahme: π sei perfekt sicher. Dann gilt nach Satz von Bayes:

$$\frac{Ws[P = p|C = c] * Ws[P = p]}{Ws[C = c]} = Ws[P = p|C = c] = Ws[P = p]$$

Daraus folgt

$$Ws[P = p|C = c] = Ws[C = c]$$

Aus $Ws[P = p|C = c] = Ws[C = c]$ folgt mit dem Satz von Bayes:

$$Ws[P = p] = Ws[P = p|C = c]$$

Damit ist nachgewiesen, dass π perfekt sicher ist. □

Der für diesen Satz nötige Satz lautet wie folgt:

Satz 3.2 (Satz von Bayes)**.**
Für zwei Ereignisse A und B, wobei $B \neq 0$ *:*

$$P_B(A) = \frac{P(A) * P_A(B)}{P(B)}$$

Hierbei bezeichnet $P_B(A)$ *die Wahrscheinlichkeit des Ereignisses A unter Vorraussetzung des Eintretens von Bedingung B. Equivalent für* $P_A(B)$ $P(A)$ *ist die sogenante Anfangswahrscheinlichkeit für das Ereignis A. Sie meint, dass das Ereignis A unabhängig zu betrachten ist.* [1]

[8] Ein Angreifer der sowohl die entschlüsselte Nachricht wie auch die verschlüsselte besitzt hat mit diesem System keine Vorteile. [9, Seite 5] Man kann auch sagen p und c sind stochastisch unabhängig. Mit dem Satz der Chiffren-Verteilung kann man folgendes nachweisen:
Satz 3.3 (Ununterscheidbarkeit von Verschlüsselung)**.** *Ein Verschlüsselungsverfahren* π *ist perfekt sicher wenn gilt:*

$$p_0, p_1 \in P, c \in C : Ws[P = p_0|C = c] = Ws[P = p_1|C = c]$$

Proof. Aus Satz 1 folgt für ein π das perfekt sicher ist:

$$Ws[P = p_0|C = c] = Ws[C = c]$$

$$\exists c \in C : Ws[C = c] = Ws[P = p_1|C = c]$$

Sei $p' \in P$ frei wählbar. Dann gilt:

$$Ws[C = c] = \sum_{p \in P} Ws[P = p|C = c] * Ws[P = p]$$

$$= Ws[P = p'|C = c] * \sum_{p \in P} Ws[P = p]$$

$$= Ws[P = p'|C = c]$$

Insgesamt folgt daraus die perfekte Sicherheit von π [9, Seite 6] □

Man kann außerdem nachweisen, dass für das perfekte Verschlüsselungsverfahren gelten muss:
Satz 3.4 (Minimale Größe des Schlüsselraumes)**.** *Annahme:* π *sei perfekt sicher. Dann gilt:* $|K| \geqslant |P|$

Proof. Beweis durch Widerspruch: Annahme: $|K| < |P|$ Für ein $c \in C$ definieren wir ein G(c) = p | p = g(c,k) mit $k \in K$. Es muss $|G(c)| \leqslant |K|$ gelten, da ein Schlüssel k genau einen Klartext p liefert. Es gilt aber $|K| < |P|$. Daraus folgt das $|G(c)| < |P|$. Somit muss es ein $p \in P$ geben für das gilt: $Ws[P = p|C = c] = 0 < Ws[P = p]$. Das würde bedeuten, dass π nicht perfekt sicher sein kann, da die Ws als > 0 definiert wurden. [9, Seite 8] □

1 http://matheguru.com/stochastik/36-satz-von-bayes.html - 25.03.2016 15:50

Mit dem Satz von Shannon können wir uns zudem noch die Verteilung der Schlüssel anschauen.
Satz 3.5 (Shannon). *Für ein Kryptosystem π = (P,C,K,f,g) mit $|P| = |C| = |K|$ gilt: π ist perfekt sicher gdw : alle $k \in K$ werden durch f gleichverteilt gewählt mit einer Wahrscheinlichkeit von $\frac{1}{|K|}$. Zudem gibt es für alle $p \in P, c \in C$ genau ein $k \in K$ sodass gilt: c = f(p,k).*

Hier beweisen wir zunächst die Gegenrichtung($\Leftarrow$):

Proof. Wir können ein $c \in C$ genau zu einem $p \in P$ entschlüsseln mit einem $k \in K$, bedeutet g(c,k) = p. Dies geschieht mit $Ws[\frac{1}{|K|}]$ gleichverteilt. Wir folgern:

$$Ws[C = c|P = p] = \frac{1}{|K|}$$

für alle $p \in P$ Hieraus folgt mit Satz 2:

$$Ws[C = c|P = p_0] = \frac{1}{|K|} = Ws[C = c|P = p_1]$$

Die Hinrichtung ($\Rightarrow$) folgt somit: Widerspruchsbeweis: Wir nehmen an: $\exists(p, c)$ mit $c \neq f(p, k)$ für alle $k \in K$ Deswegen gilt: $Ws[P = p|C = c] = 0 < Ws[P = p]$. Dies steht im Widerspruch zu unserer Definition von einem perfekt sicherem System.

Nehmen wir anders herum an: $\exists(p, c)$ mit $c = f(p, k)$ für mehrere $k \in K(\geqslant 2)$. Dann gilt: $\exists(p', c')$ mit $c' \neq f(p', k)$ für alle $k \in K$.

Dies erzeugt einen Widerspruch wie oben. Daraus folgern wir: Es gibt für jedes feste $p \in P, c \in C$ genau ein $k \in K$ für das gilt: f(p,k) = c. Im weiteren ergibt sich für alle p, p' : $Ws[K = k_p] = Ws[C = c|P = p] = Ws[C = c|P = p'] = Ws[K = k'_p] \rightarrow Ws[K = k] = \frac{1}{|K|}$ für alle $k \in K$ [9, Seite 9] □

Intuitiv bedeutet perfekte Sicherheit also:
Der Schlüssel, der zur Verschlüsselung der Nachricht verwendet wird muss mindestens so lang sein wie die Nachricht selbst. Der Schlüssl muss zudem noch perfekt gleichverteilt von der Verschlüsselungsfunktion gewählt werden. Der Angreifer hat keinerlei Vorteile durch das Wissen um ein Klartext - verschlüsselter Text Paar. Das schließt auch sogenannte Meet-In-The-Middle Attacken aus, auf die später noch eingegangen wird. Mit all diesen nicht gerade leicht zu erfüllenden Bedingungen gibt es nicht viele Beispiele aus der Vergangenheit, von Verschlüsselungsverfahren die die Kriterien der perfekten Sicherheit erfüllten.

3.2.2 *Vernam'sches One-Time-Pad*

Eines dieser wenigen Beispiel ist das Vernam'sche One-Time-Pad, eine 1918 von Gilbert Vernam entwickelte Verschlüsselungsmethode, die zum verschlüsseln des roten Telefons im Kalten Krieg zwischen dem US-Präsidenten und dem sowjetischen Generalsekretär genutzt wurde. Die Besonderheit ist, dass der Schlüssel genauso lang wie die zu verschlüsselnde Nachricht war. Zur Ver-und Entschlüsselung wurden Buchstaben nach einer Tabelle addiert.

+	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Abbildung 2: One-Time Pad Tabelle[10]

Addiert wurde mit einer zufälligen Buchstabenfolge die beiden Seiten bekannt war. [10] Mathematisch wurde das One-Time-Pad definiert durch ein Kryptosystem $\pi = (P, C, K, f, g)$ wobei $P = C = K = \{0,1\}^l$ für das gilt:

$$\exists k \in K, \exists p \in P : c \in C = f(p,k) = p \oplus k$$

$$\exists k \in K, \exists c \in C : p \in P = g(c,k) = c \oplus k$$

h Annahme: Das One-Time-Pad ist ein perfekt sicheres Verfahren.

Proof. Da $C = P \oplus K$ gilt: $\forall p_0, p_1 \in Pundc \in C : Ws[C = c|P = p_0] = Ws[P \oplus K = p_0] = Ws[K = p_0 \oplus c] = \frac{1}{2^l} = Ws[C = c|P = p_1]$ □

Dies entspicht dem Satz über Ununterscheidbarkeit von Verschlüsselung. Somit ist das One-Time-Pad perfekt sicher. [9, Seite 7]
Das Beispiel zeigt gut, wie wenig praktikabel eine Verschlüsselung mit perfekter Verschlüsselung im Alltag ist. Ein so langer Schlüssel macht nicht nur das Ver- und Entschlüsseln langsam, sondern fördert auch das Auftreten von Fehlern. Wenn beim One-Time-Pad die zufälligen Buchstabenfolgen nicht übereinstimmten oder etwa falsch gelesen wurden, konnten beide Seiten nicht mehr kommunizieren und es mussten neue Folgen ausgetauscht werden, was wiederum anfällig für Fehler und Angriffe von 3ten Parteien war. Somit kann das Ziel bei der Konstruktion eines Verschlüsselungsalgorithmus nie sein, ihn perfekt sicher zu machen, sondern nur sicher genug.
In der Realität haben die Angreifer auf eine Verschlüsselung nie unbegrenzte Resourcen zur Verfügung. Somit ist ein Schlüssel, der sehr viel kürzer als die Länge des Klartextes ist, oftmals ausreichend. Auch ist die Wahl eines Algorithmus, der die Schlüssel absolut gleichverteilt wählt, sehr schwierig und nur bei einem riesigem Schlüsselraum realistsisch. Ein solches Verschlüsselungsverfahren zu bauen und zu verwalten ist nicht praktisch.

3.3 Diffusion und Konfusion - Alexej Rotar

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. [11, S. 52] Von Shannon wurden zwei allgemeine Methoden entwickelt, die solche Analysen erschweren.

3.3.1 *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. [11, S. 53-54] 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 $\frac{1}{2}$. [1, S. 45]

3.3.2 *Konfusion*

Konfusion veschleiert den Zusammenhang zwischen dem Geheimtext und dem Schlüssel. So wird ein statistischer Angriff ebenfalls erschwert. Denn so muss unter Umständen ein komplexes Gleichungssystem gelöst werden, um einen Schlüssel zu ermitteln. Damit ein Verfahren konfus ist, sollte sich möglichst jeder Teil des Schlüssels auf die Geheimnachricht auswirken. [11, S. 54-55]

3.3.3 *Konstruktion einer sicheren Chiffre*

Eine Möglichkeit, ein Verfahren zu konstruieren, welches diffus und konfus ist, ist es ein Verfahren zu iterieren, bei dem sich Substitutionen mit Permutationen abwechseln. Damit man dasselbe Verfahren mehrfach anwenden kann, ist es notwendig, dass es sich um eine *endomorphe* Chiffre handelt. Das heißt, es müssen Bild- und Urbildmenge diesselbe sein. [2, S. 111]

4 BLOCKCHIFFREN – ALEXEJ ROTAR

4.1 Allgemein

Bei Verfahren wie beispielsweise der Caesar-Chiffre wird jedes Zeichen einzeln verschlüsselt. Bei manchen Verfahren werden dagegen ganze Blöcke verschlüsselt. Dann spricht man von *Blockchiffren*. Dabei wird eine Nachricht P in mehrere Blöcke der Länge l aufgeteilt. Sollte l kein Teiler der Länge von P 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. [1, S. 33]

4.2 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.

4.2.1 *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

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

Dann ist die Feistel-Chiffre definiert durch

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

Nun wird zunächst 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. [1, S. 51] 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:

4.2.2 *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.*

2 Hier wurde die Funktion g bewusst weggelassen, da für die interne Blockchiffre keine Entschlüsselungsfunktion notwendig ist.

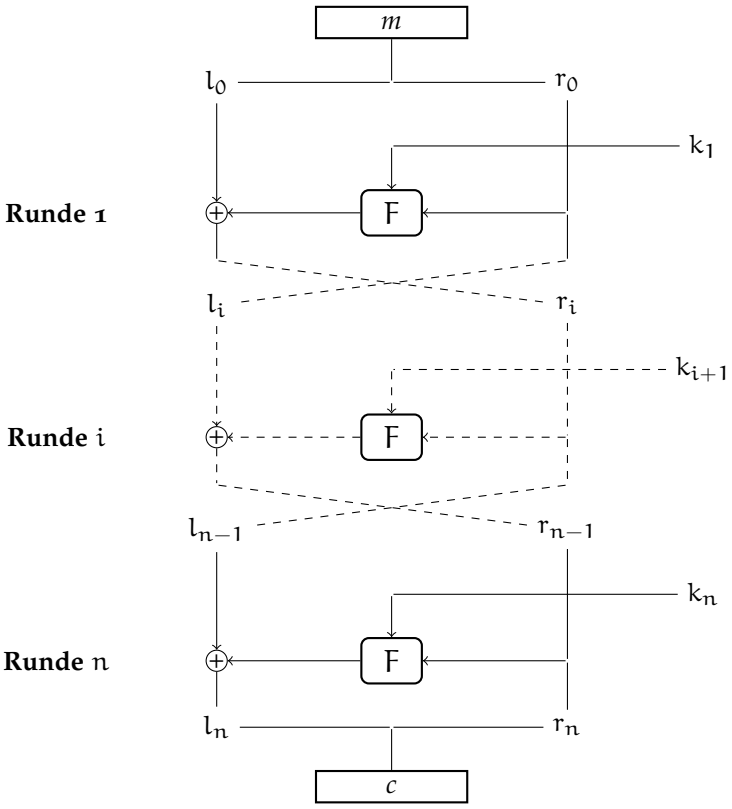


Abbildung 3: Ablauf einer Feistel-Chiffre [2, S. 128]

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 ermittelt werden. [1, S. 51]

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 $i = N - j$. Außerdem 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 – ALEXEJ ROTAR

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.[1, S. 51-52] 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. [1, S. 52] Die einzelnen Funktionen sind folgendermaßen definiert:

5.2.1 *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 [1, S. 52]

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. [1, S. 52]
Da sie sich als Matrixmultiplikation darstellen lässt, handelt es sich bei der Expansion um eine lineare Abbildung. Sie erhält außerdem weitgehend die Struktur des Blocks und kann somit nicht der Diffusion dienen. Allerdings ist sie notwendig, um die Nachricht auf die Größe eines Rundenschlüssel zu bringen, der anschließend damit addiert wird.

5.2.2 *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 [1, S. 52]

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. [2, S. 138] An einem einfachen Beispiel sieht man schnell, dass die Substitution keine lineare Abbildung ist.

5.2.3 *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 [2, S. 139]

An der Tabelle sieht man, dass der Block durch die Permutation zerstreut wird. Somit wirkt sich jeder Teil der Nachricht an vielen Stellen der Geheimnachricht aus, wodurch das Verfahren diffus wird. Ähnlich wie die Expansion, ist auch die Permutation eine lineare Abbildung.

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. [2, S. 133] 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. [2, S. 141] 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.

Als einzige nicht-lineare Funktion sorgt die Substitution auch für die nicht-Linearität des gesamten Verfahrens und ist damit ein entscheidender Faktor für dessen Sicherheit. Denn lineare Abbildungen lassen sich sehr gut analysieren. [1, S. 21] 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. [1, S. 52]

6 KRYPTOANALYSE – ALEXEJ ROTAR

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. [12, S. 24]
Es wurden für einen Angriff gegen den DES zwei Methoden vorgestellt. 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. [2, S. 153] 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

$$k^{(1)} = e^{(1)} \oplus E^{(1)}(x) \tag{5}$$

Wobei damit jeweils die ersten 6 Bits 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. [2, S. 160] 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. [2, S. 139] 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. [2, S. 155-156] Jedes dieser Paare ist potentiell

richtig und bildet mit Formel 5 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. [2, S. 160]
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. [2, S. 156]

6.3 Bestimmung der Differenzen

Man kann mit unterschiedlichen Methoden die Ein- und Ausgabedifferenzen des DES mit bis zu fünf Runden bestimmen. Ab sechs Runden ist das nicht mehr mit Sicherheit möglich. Stattdessen kann man nur noch mit gewissen Wahrscheinlichkeiten die Differenzen vorhersagen. Zunächst sollen nun einige Hilfsmittel definiert werden, um anschließend gewisse Aussagen über die Differenzen und Schlüssel treffen zu können.
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

$$P(F, \Delta x \to \Delta y) := \frac{|\{(x, x', k) | x \oplus x' = \Delta x \wedge F(k, x) \oplus F(k, x') = \Delta y\}|}{2^{32} \cdot 2^{48}} [2, S.164 - 165]$$

Satz 6.1. *Zwischen den eben definierten Wahrscheinlichkeiten gilt 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

$$p^{\Gamma} := \prod_{i=1}^{n} P(F, \Delta x_i \to \Delta y_i) [2, S.166 - 167]$$

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 wird. [2, S. 172-174]

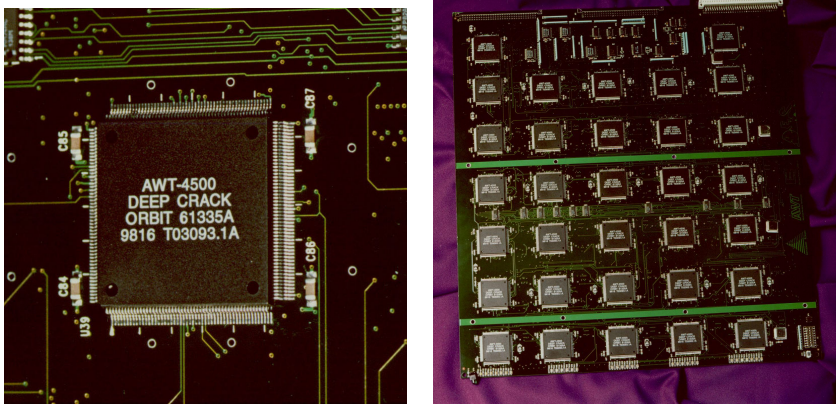
7 SCHLUSS – SIMON STERNSDORF

7.1 Kritik an DES

Schon recht früh nach Einführung des DES-Algorithmus als Standard-Verschlüsselung in vielen Bereichen des Öffenlichen Lebens kam Kritk auf. So wurde bereits früh, im Jahr 1975, vor der Möglichkeit eines Brute-Force Angriffs gewarnt, der aufgrund der recht kurzen Schlüssellänge gut möglich wäre. Im speziellen Martin Hellman und Whitfield Diffie, die Erfinder des Konzepts der Public-Key Verschlüsselung, warnten vor möglichen Angriffen von Regierungsorganisationen, im speziellen Geheimdienste. Diese hätten Zugriff auf die für damalige Verhältnisse recht hohe Rechenleistung. [4]

7.2 Brute-Force

Ein Brute-Force-Angriff ist die einfachste und gewaltsamste Art eine Verschlüsselung zu knacken. Dabei wird von einem Computer systematisch jede Kombination aus Zahlen, Buchstaben und Sonderzeichen bis zu einer zuvor festgelegten Länge durchprobiert. Je kürzer dabei die Länge des verwendeten Schlüssels ist desto schneller ist das Verfahren. Zudem hängt es von der verwendeten Rechenpower des Angreifers ab. Ein höherer Einsatz von Rechenleistung erhöht die Geschwindigkeit der Suche. [13]
Moderne Verschlüsselungen setzen deswegen heutzutage auf Schlüssel, die sehr viel länger sind als die 56 Bit des DES-Algorithmus. Der weit verbreitete Verschlüsselungsalgorithmus AES setzt auf Schlüssellängen bis zu 256 Bit. [14]



(a) Der verwendete Cracker-Chip. **(b)** Ein Chip-Board

Abbildung 4: DES-Cracker Chip [15]

7.3 Der DES-Cracker

Im Jahr 1998 gelang es schließlich der Electronic Frontier Foundation, eine Organisation die sich für digitale Grundrechte und Privatssphäre einsetzt, mithilfe einer selbst gebauten "Crackers" den DES-Algorithmus mittels Brute-Force zu knacken. Die Maschine kostete nur 250.000 Dollar und benötigte nur 3 Tage um diese Aufgabe zu bewerkstelligen. Die EFF wollte damit der Öffentlichkeit aufzeigen, wie leicht der immer noch in weiten Teilen der Industrie eingesetzte Verschlüsselungsalgorithmus zu knacken war. 1999 gelang es mit dem DES-Cracker und einem weltweitem Computernetzwerk namens Distributed.Net eine DES-verschlüsselte Nachricht in nur 22 Stunden zu knacken. Die Computer schafften es zusammen auf eine Geschwindigkeit von 245 Millarden Schlüsseltests pro Sekunde. Das ganze geschah im Rahmen eines Wettbewerbs, um die von der US-Regierung immer noch bestehende Behauptung zu widerlegen, dass es Jahre brauchen würde und Rechenleistung im Wert von mehreren Millionen Dollar um eine DES-Nachricht mittels Brute-Force zu entschlüsseln. [15]

7.4 Entwickelte Alternativen

Die EFF schlug schon 1994 dem X9 Kommittee, das für die Empfehlung von neuen Verschlüsselungsalgorithmen verantwortlich war als Alternative triple-DES oder kurz 3DES vor. Damit stellten sie sich klar gegen die NSA und deren Vorschlag eines speziellen Verschlüsselungschips namens Clipper, der einen eingebauten "Backdoor" enthielt, um so der NSA Zugriff auf verschlüsselte Kommunikation zu geben. Ein Backdoor ist eine Hintertür in einem eigentlich als sicher geltendem System. In der Öffentlichkeit nichts von dieser Sicherheitslücke bekannt und sie wird nur von bestimmten Organisationen genutzt. Dabei wurde eine Backdoor im Gegensatz zu einem Exploit durch einen Programmfehler speziell beim Entwerfen des Programms eingebaut. [16] Die EFF schlug dabei 3DES aus den folgenden Gründen vor:

- Grundsätzlich ist DES, auf dem 3DES basiert ein kryptologisch sicherer Algorithmus

- Es eliminiert (für die damalige Zeit) die Möglichkeit einer Brute-Force-Attacke durch den viel längeren Schlüssel, der verwendet wird. Die Schlüssellänge wird zu DES verdoppelt.
- Es ist leicht in bereits vorhandene Systeme für DES zu integrieren

[17]

7.5 Ungeeignete Alternative: 2DES

Da man mit 3DES die Schlüssellänge nur verdoppelt kann man sich fragen, warum man nicht einfach nur 2DES verwendet. Hierbei würde man mit 2 DES Schlüsseln einen Klartext zunächst mit dem 1ten Schlüssel und danach diesen verschlüsselten Text mit dem 2ten Schlüssel verschlüsseln. Da DES keine Gruppe ist würde dies tatsächlich nicht nur einem neuen Schlüssel entsprechen. Allerdings ist dieses Verschlüsselungsverfahren anfällig für die sogenannte Meet-in-the-middle Attacke. [18] Die Meet-in-the-middle Attacke kann nur bei einem Verschlüsselungsalgorithmus mit einem sogenannten Intermediate State funktionieren. Eine kleine Darstellung einer Meet-in-the-Middle Attacke:

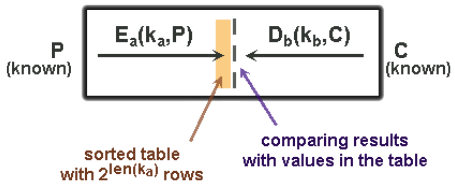


Abbildung 5: Darstellung einer Meet-In-The-Middle Attacke[19]

Wir brauchen dabei einen Klartext P und den dazugehörigen verschlüsselten Text C. Wir haben nun also:

$$P \rightarrow f(P, k_1) \rightarrow f(f(P, k_1), k_2) \rightarrow C$$

Der Trick ist nun den Zustand $f(P, k_1)$ zu erreichen. Dabei wird zuerst ganz klassisch mit Brute-Force der Klartext P mit allen möglichen Schlüsseln (2^{56} Bit) verschlüsselt und die Resultate gespeichert. Danach macht man das gleiche mit C und der Entschlüsselungsfunktion.

Es gilt also: $P \rightarrow f(P, k_1) = d(C, k_2) \leftarrow C$

Man vergleicht immer mit dem Resultat der Verschlüsselung von P. Es kann dabei mehrere Paare geben die zusammen passen, aber endlich viele. Diese kann man dann ausprobieren ob sie auch bei anderen verschlüsselten Texten funktionieren um so die richtigen 2 Schlüssel herauszufinden. [19]

Größter limitierender Faktor war damals noch der hohe Speicherplatzverbrauch. Man brauchte die Kapazität die gesamten Intermediate Texte zu speichern. Das ist heutzutage kein Hindernis mehr. [20]
Wenn man das Verfahren in 2 oder mehr einfachere Verfahren aufteilen kann, so ist es möglich eine sogeannte “mehrdimensionale MitM Attacke“

durchzuführen. Dies ist vor allem bei Blockchiffren ein Problem, die auf kleinen Datenblöcken mit sehr großen Schlüsseln agieren. Die Schlüssel werden hierbei effektiv aufgeteilt.

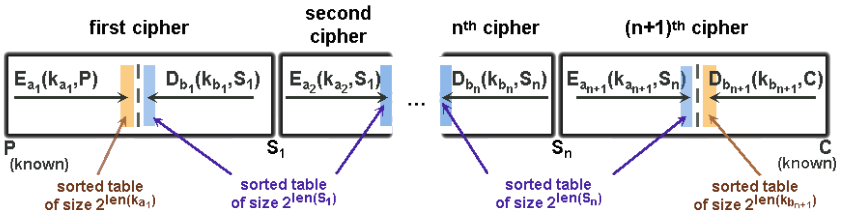


Abbildung 6: Darstellung einer mehrdimenionalen MitM Attacke[19]

Schlussendlich bekommt man trotz der 2 verwendeten Schlüssel bei 2DES nicht eine resultierende Schlüssellänge von 2^{112} Bit, statdessen braucht man mit Brute-Force nur doppelt so lange um die Nachricht zu entschlüsseln, sollte man im Besitz eines Klartext- verschlüsselter Text Paares sein. Das enspricht 2^{57} Bit. Damit war 2DES nicht zukunftstauglich und wurde nicht in Betracht gezogen als Alternative für DES. [18]

7.6 triple-DES

3DES wurde im Standart X9.52 festgehalten und wurde zur neuen empfohlenen Verschlüsselungsmethode in der FIPS, der Federal Information Processing Standard. Dieser regelt die Standards, die Firmen erfüllen müssen um Verträge mit der US-Regierung zu schließen. Dies trug maßgeblich zu Verbreitung des 3DES in viele mit der Regierung zusammenhängenden Wirtschaftszweige bei. Zugleich wurde Single-DES oder DES nur noch für wenige, speziell zugelassene "legacy systems" erlaubt. Das sind Systeme die zu alt und groß waren, und die eher intern genutzt wurden.
Zudem wurde die langsame Umstellung auf AES empfohlen, da dieser auch von der NSA speziell authorisiert war. Zunächst wollte man aber die hardwarebasierten DES-Systeme weiter nutzen, deswegen war 3DES ein logischer Schritt. [21] 3DES basiert auf dem DES Algorithmus. Um etwas mit 3DES zu verschlüsseln nimmt man 2 DES Schlüssel mit jeweils 56 Bit und wendet sie wie folgt an:

- Verschlüssele den Klartext mit dem ersten Schlüssel
- Entschlüssle die Nachricht mit dem zweiten Schlüssel
- Verschlüssle die Nachricht nochmal mit dem ersten Schlüssel

Diese besondere Anrt der Anwendug der 2 Schlüssel bedeutet eine Verlängerung des insgesamten Schlüssel auf 112 Bit. Dadurch ist er sehr viel schwerer mit Brute-Force zu knacken. [4] Es hat keinerlei Auswirkungen auf die Sicherheit des Verfahrens ob man in Schritt 2 und 3 ver-oder entschlüsselt.

Aber in der oben genannten Reihenfolge ist es üblich. Man kann nur nicht immer den gleichen Schlüssel benutzen. [22]

Ein Nachteil von 3DES im Gegensatz zu anderen Verschlüsselungsverfahren ist seine Performanz. Da auf Hardwareverschlüsselung optimiert, ist die dreimalige Anwendung des DES-Algorithmus in Software form deutlich langsamer als von vergleichbaren Algorithmen mit ähnlich hoher Sicherheit. Deswegen ist 3DES im Privatgebrauch auch kaum verbreitet. Hier hat sich vor allem AES durchgesetzt wegen seiner sehr reduzierten Codebasis. [23]

REFERENCES

[1] C. Karpfinger and H. Kiechle, *Kryptologie*. Vieweg+Teubner, 1 ed., 2010.

[2] M. Miller, *Symmetrische Verschlüsselungsverfahren*. B. G. Teubner GmbH, 1 ed., 2003.

[3] C. Stobitzer, "Caesar verschlüsselung/ caesar chiffre." ttp://www.kryptowissen.de/caesar-chiffre.html, 2013. Zugriff: 07.03.2016 13:50.

[4] J. O. Grabbe, "The des algorithm illustrated," *Laissez Faire City Times*, vol. 2, no. 28. Zugriff: 07.03.2016 13:50.

[5] R. Spier, "Der data encryption standard (des)." http://www-lehre.informatik.uni-osnabrueck.de/~rspier/referat/internet/DES-Algorithmus.html, 2001. Zugriff: 07.03.2016 13:50.

[6] J. Buchmann, *Einführung in die Kryptologie*. Springer-Verlag Berlin Heidelberg, 4te, erweiterte auflage ed., 2008.

[7] A. May, "Symmetrische kryptographie." http://www.cits.rub.de/imperia/md/content/may/ws1516/krypto_i.pdf, 2015. Zugriff: 09.03.2016 15:05.

[8] A. Schneider, "Satz von bayes." http://www.mathebibel.de/satz-von-bayes. Zugriff: 25.03.2016 14:20.

[9] A. May, "Prinzipien der modernen kryptographie - sicherheit." http://www.cits.rub.de/imperia/md/content/may/0910/ws0910/krypto1ws09/02_perfekt.pdf, 2009. Zugriff: 09.03.2016 15:30.

[10] F. Zander and B. Reiß, "Das one-time-pad." http://www.mathe.tu-freiberg.de/~hebisch/Praktikum11-4/Seite2.html, Mai 2011. Zugriff: 09.03.2016 16:00.

[11] C. Shannon, "Communication theory of secrecy systems." http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf. Zugriff: 28.03.2016 09:34.

[12] W. Ertel, *Angewandte Kryptographie*. Carl Hanser Verlag München, 3 ed., 2007.

[13] K. Lipinski, H. Lackner, O. P. Laué, G. Kafka, A. Niemann, E. Raasch, B. Schoon, and A. Radonic, "Brute-force-angriff." http://www.itwissen.info/definition/lexikon/

Brute-Force-Angriff-brute-force-attack.html, 2015. Zugriff: 11.03.2016 17:20.

[14] "Advanced encryption standard (aes)," 2001. Zugriff: 29.03.2016 17:30.

[15] E. F. Foundation, ed., *Cracking DES*. O'Reilly Media, 1998. Zugriff: 12.03.2016 - 15:20.

[16] K. Lipinski, H. Lackner, O. P. Laué, G. Kafka, A. Niemann, E. Raasch, B. Schoon, and A. Radonic, "Backdoor." http://www.itwissen.info/definition/lexikon/Backdoor-backdoor.html, 2012. Zugriff: 12.03.2016 - 15:40.

[17] "Effector online," vol. 07, no. 14, 1994. Zugriff: 12.03.2016 15:40.

[18] C. Christensen, "Double des." http://www.nku.edu/~christensen/3DES.pdf. Zugriff: 25.03.2016 - 14:30.

[19] C. Kowalczyk, "Meet-in-the-middle attack." http://www.crypto-it.net/eng/attacks/meet_in_the_middle.html, 2013. Zugriff: 28.03.2016 13:00.

[20] M. Rouse, "Meet-in-the-middle attack definition." http://internetofthingsagenda.techtarget.com/definition/meet-in-the-middle-attack, 2010. Zugriff: 25.03.2016 15:20.

[21] "Fips publication 46-3," 1999. Zugriff: 12.03.2016 16:50.

[22] K. Lipinski, H. Lackner, O. P. Laué, G. Kafka, A. Niemann, E. Raasch, B. Schoon, and A. Radonic, "triple-des." http://www.itwissen.info/definition/lexikon/triple-DES-3DES-Dreifach-DES.html, 2007. Zugriff: 12.03.2016 16:30.

[23] B. Karadeniz, "Kryptographie im internet - verschlüsselungsverfahren." http://www.netplanet.org/kryptografie/verfahren.shtml. Zugriff: 12.03.2016 16:30.