1 Komposition von Chiffren

Ein erstes Konstruktionsprinzip für starke Chiffren, das auch in der klassischen Kryptographie oft angewendet wurde, ist die Nacheinander-Ausführung von Chiffrierschritten. Solche Verknüpfungen können zu beachtlicher Verstärkung der Sicherheit führen; selbstverständlich ist das aber nicht – und es stimmt auch nicht immer. In diesem Abschnitt werden einige grundlegende Überlegungen dazu vorgestellt.

Dabei wird unter einer

Mehrfach-Chiffre die Nacheinanderausführung des gleichen Verfahrens, aber mit verschiedenen Schlüsseln,

Kaskade die Nacheinanderausführung verschiedener Verfahren

verstanden.

1.1 Mehrfach-Chiffren und Gruppenstruktur

Mehrfach-Chiffren

Sei $F = (f_k)_{k\in K}$ eine Chiffre über dem Alphabet Σ, also $f_k\colon \Sigma^* \longrightarrow \Sigma^*$ die zugehörige Verschlüsselungsfunktion für jeden Schlüssel $k \in K$. Die gesamte Menge von Verschlüsselungsfunktionen wird mit

$$\tilde{F} = \{f_k \mid k \in K\} \subseteq \mathrm{Abb}(\Sigma^*, \Sigma^*)$$

bezeichnet.

Der Schlüsselraum wird wesentlich vergrößert, nämlich von K zu $K \times K$, durch die Bildung der **Zweifach-Chiffre**

$$F^{(2)} = (f_h \circ f_k)_{h,k\in K}$$

Natürlich kann man ebenso die Dreifach-Chiffre $F^{(3)}$, ..., die n-fach-Chiffre $F^{(n)}$ bilden. Sinnvoll ist das alles nur, wenn

(A) $\tilde{F}$ keine Halbgruppe ist.

Ist nämlich $\tilde{F}$ eine Halbgruppe, so gibt es zu zwei Schlüsseln $h, k \in K$ stets einen weiteren Schlüssel $x \in K$ mit $f_h \circ f_k = f_x$. Durch Komposition entstehen also keine neuen Verschlüsselungsfunktionen, sie ist eine „illusorische Komplikation“.

Noch besser ist, wenn

(B) $\tilde{F}$ eine möglichst große Unter-Halbgruppe von $\mathrm{Abb}(\Sigma^*, \Sigma^*)$ erzeugt.

Und das beste, was man hier erreichen kann, ist:

(C) Die Abbildung $K \times K \longrightarrow \widetilde{F^{(2)}} \subseteq \mathrm{Abb}(\Sigma^*, \Sigma^*)$ ist injektiv;

das kann man für einen endlichen Schlüsselraum K auch so ausdrücken:

(C') $\#\widetilde{F^{(2)}} = \#\{f_h \circ f_k \mid h, k \in K\} = (\#K)^2$.

Die Gruppen-Eigenschaft von Blockchiffren

Für Blockchiffren gilt im wesentlichen auch die Umkehrung; das folgt aus:

Eine Blockchiffre ist durch die Wirkung auf einem Σ^r (für einen gegebenen Exponenten r) eindeutig festgelegt. (Um die Details der Fortsetzung auf Zeichenketten beliebiger Länge und des Auffüllens oder „Padding“ von zu kurzen Ketten auf Blocklänge kümmern wir uns im Moment nicht.)

Eine Blockchiffre heißt **längentreu**, wenn sie Σ^r in sich abbildet. Insbesondere ist dann $\tilde{F}$ auf natürliche Weise Teilmenge der symmetrischen Gruppe $\mathcal{S}(\Sigma^r)$, also endlich, und man kann den Schlüsselraum K ohne Einschränkung als endlich annehmen. Für solche Blockchiffren ist die Halbgruppen-Eigenschaft (also die Negation von (A) oben) zur Gruppen-Eigenschaft äquivalent. Das folgt aus dem bekannten einfachen:

Hilfssatz 1 *Sei G eine endliche Gruppe, $H \leq G$ eine Unter-Halbgruppe, d. h., $H \neq \emptyset$ und $HH \subseteq G$. Dann ist H Gruppe, insbesondere $\mathbf{1} \in H$.*

Beweis. Jedes $g \in G$ hat endliche Ordnung, $g^m = \mathbf{1}$. Ist nun $g \in H$, so $\mathbf{1} = g^m \in H$ und $g^{-1} = g^{m-1} \in H$. $\diamond$

Daher ist bewiesen:

Satz 1 *Sei F eine längentreue Blockchiffre über einem endlichen Alphabet. Dann sind folgende Aussagen äquivalent:*

(i) *Zu je zwei Schlüsseln $h, k \in K$ gibt es $x \in K$ mit $f_h \circ f_k = f_x$.*

(ii) *Die Menge $\tilde{F}$ der Verschlüsselungsfunktionen ist eine Gruppe.*

Anmerkung

Die Wahrscheinlichkeit, dass zwei zufällige Elemente der symmetrischen Gruppe $\mathcal{S}_n$ bereits die ganze Gruppe $\mathcal{S}_n$ oder wenigstens die alternierende Gruppe $\mathcal{A}_n$ erzeugen, ist

$$> 1 - \frac{2}{(\ln \ln n)^2} \quad \text{für große } n.$$

Für $n = 2^{64}$, einen typischen Wert bei Blockchiffren, ist diese untere Schranke ≈ 0.86. Eine nicht ganz ungeschickt gewählte Blockchiffre wird also sehr wahrscheinlich die volle oder wenigstens halbe Permutationsgruppe auf den Blöcken erzeugen. Trotzdem ist der konkrete Nachweis davon oft schwer. Jedenfalls kann man davon ausgehen, dass eine Mehrfach-Chiffre „in der Regel“ stärker als die Einfach-Chiffre ist.

Quelle: John Dixon, *The probability of generating the symmetric group.* Mathematische Zeitschrift 110 (1969), 199–205.

1.2 Beispiele für Mehrfach-Chiffren

Beispiele für Gruppen

Jeweils eine Gruppe bilden die folgenden längentreuen Blockchiffren:

- die Verschiebechiffren über Σ bezüglich einer Gruppenstruktur auf Σ,
- die monoalphabetischen Substitutionen über Σ,
- die BELASO-Chiffren fester Periode,
- die Block-Transpositionen fester Länge.

DES

DES ist eine Blockchiffre auf $\mathbb{F}_2^{64}$ mit Schlüsselraum $\mathbb{F}_2^{56}$. CAMPBELL und WIENER haben gezeigt (CRYPTO 92), dass DES die alternierende Gruppe der Ordnung 2^{64} erzeugt.

Historische Beispiele

Die Komposition einer polyalphabetischen Chiffre der Periode l mit einer der Periode q hat die Periode $\mathrm{kgV}(l, q)$. **Anwendung:** Schlüsselerzeugermaschinen, wie sie in Kapitel I am Ende des Abschnitts über Chiffrierzylinder sowie bei den historischen Rotormaschinen kurz erwähnt wurden.

Weiteres historisches Beispiel: Die doppelte Spaltentransposition, die wesentlich schwerer zu brechen ist, als die einfache.

Komposition von BELASO-Chiffren

Die Komposition zweier BELASO-Chiffren der Perioden l und q hat zwar die Periode $\mathrm{kgV}(l, q)$, also im wesentlichen das Produkt lq, die Sicherheit entspricht aber höchstens der Summe $l + q$, wenn man einen Angriff mit bekanntem Klartext berücksichtigt:

Aus einem bekannten Klartext der Länge $l + q$ gewinnt man $l + q$ lineare Gleichungen für die ebensovielen Unbekannten, aus denen die beiden Schlüssel insgesamt bestehen. Sei dazu o. B. d. A. $l < q$. Dann haben wir die Situation

Klartext	a_0	a_1	$\ldots$	a_{l-1}	a_l	$\ldots$	a_{q-1}	$\ldots$
Schlüssel 1	h_0	h_1	$\ldots$	h_{l-1}	h_0	$\ldots$	$\ldots$	$\ldots$
Schlüssel 2	k_0	k_1	$\ldots$	k_{l-1}	k_l	$\ldots$	k_{q-1}	$\ldots$
Geheimtext	c_0	c_1	$\ldots$	c_{l-1}	c_l	$\ldots$	c_{q-1}	$\ldots$

Wir haben also eine BELASO-Chiffre mit dem Gesamtschlüssel

$$(h_0 + k_0, h_1 + k_1, \ldots)$$

und der Periode $\mathrm{kgV}(l, q)$.

Nehmen wir nun bekannten Klartext der Länge $l+q$ an, etwa o. B. d. A. $(a_0, \ldots, a_{l+q-1})$. Dann haben wir das folgende lineare Gleichungssystem für die $l+q$ Unbekannten $h_0, \ldots, h_{l-1}, k_0, \ldots, k_{q-1} \in \mathbb{Z}/n\mathbb{Z}$:

$$\begin{aligned}
h_0 + k_0 &= c_0 - a_0, \\
h_1 + k_1 &= c_1 - a_1, \\
&\vdots \\
h_{l-1} + k_{l-1} &= c_{l-1} - a_{l-1}, \\
h_0 + k_l &= c_l - a_l, \\
&\vdots \\
h_{l+q-1 \bmod l} + k_{l+q-1 \bmod q} &= c_{l+q-1} - a_{l+q-1}.
\end{aligned}$$

Dieses ist mit Sicherheit nicht eindeutig lösbar: Addiert man einen festen Wert x zu allen h_i und subtrahiert x von allen k_j, so hat man ebenfalls eine Lösung. Daher kann man zunächst der Einfachheit halber $h_0 = 0$ annehmen; sollten die Schlüssel nicht zufällig gewählt, sondern aus Schlüsselwörtern gebildet sein, kann man am Ende mit einer einfachen „Caesar-Exhaustion“ wie ganz am Anfang von Kapitel I die „wahren“ Schlüssel ermitteln. Für die Dechiffrierung tun es auch die verschobenen Schlüssel. Und da wir eine Unbekannte weniger haben, reichen im allgemeinen sogar $l+q-1$ bekannte Klartext-Zeichen. Dies soll hier nicht weiter ausgeführt werden; der Interessierte möge die folgende Aufgabe lösen.

Übungsaufgabe

Gegeben sei der Geheimtext.

```
CIFRX KSYCI IDJZP TINUV GGKBD CWWBF CGWBC UXSNJ LJFMC
LQAZV TRLFK CPGYK MRUHO UZCIM NEOPP LK
```

Für einen Angriff mit bekanntem Klartext

- bestehe die berechtigte Vermutung, dass der Klartext mit „Sehr geehrter ...“ beginnt,
- seien mit Trial & Error plausible Schlüssellängen ausprobiert; es sei $42 = 6 \times 7$ an der Reihe, um zu testen, ob eine doppelte Belaso-Verschlüsselung mit den Schlüssellängen 6 und 7 vorliegt.

(Hierzu braucht man nicht alle Schlüssellängen und Kombinationen durchzuprobieren; die Koinzidenzanalyse ist wegen der Kürze des Textes nicht allzu spezifisch, reicht aber, um alle bis auf wenige plausible Schlüssellängen auszuschließen.)

1.3 Kryptoanalyse von Zweifach-Chiffren

Der Treffpunkt-Angriff

Diese Attacke auf Zweifach-Chiffren wurde 1981 von MERKLE und HELLMAN unter dem Namen „Meet in the Middle“ vorgestellt; sie ist nicht mit einem „Man in the Middle“-Angriff auf kryptographische Protokolle zu verwechseln.

Betrachtet wird die Komposition von zweimal der gleichen Chiffre mit verschiedenen Schlüsseln:

$$\begin{array}{ccccc} \Sigma^* & \xrightarrow{f_k} & \Sigma^* & \xrightarrow{f_h} & \Sigma^* \\ a & \mapsto & b & \mapsto & c. \end{array}$$

Sei ein bekanntes Klartext-Geheimtextpaar (a, c) gegeben. Dann bildet der Angreifer zwei Tabellen:

- alle $f_k(a)$, $k \in K$,
- alle $f_h^{-1}(c)$, $h \in K$,

und vergleicht diese. Jede Übereinstimmung ergibt ein mögliches Schlüsselpaar $(h, k) \in K^2$, das weiter getest werden kann, etwa an einem weiteren bekannten Klartext.

Aufwand

Benötigt werden für diesen Angriff

- $2 \cdot \#K$ Verschlüsselungsoperationen (*nicht etwa* $(\#K)^2$!),
- $2 \cdot \#K$ Speicherplätze,

wobei die Zahl der nötigen Speicherplätze durch die Bemerkung halbiert wird, dass man nur eine der beiden Tabellen abspeichern muss.

Speichergrößen werden bekanntlich so bezeichnet:

2^{10}	2^{20}	2^{30}	2^{40}	2^{50}
Kilo	Mega	Giga	Tera	Peta

Dabei ist der Speicherbedarf noch mit der Größe eines Blocks des Verschlüsselungsverfahrens, etwa 64 Bit = 8 Byte, zu multiplizieren.

Man sieht, dass man schon mit 50-Bit-Schlüsseln in Größenbereiche kommt, die mit heutigen Speichertechniken nicht realisierbar sind. Da es bei der Kryptoanalyse allerdings mehr auf den Zeit- als auf den Speicherbedarf ankommt, ist die allgemeine Aussage gerechtfertigt:

> *Eine Zweifach-Chiffre ist nicht wesentlich sicherer als die zugrundeliegende Einfach-Chiffre.* Insbesondere ist die Bitlänge für die exhaustive Schlüsselsuche bei weitem nicht verdoppelt.

Fehlalarme

Eine Frage ist bei der Analyse offen geblieben: Wieviele der beim Tabellenabgleich gefundenen Übereinstimmungen führen zu einem falschen Schlüsselpaar? D. h., wie groß ist die Wahrscheinlichkeit eines Fehlalarms?

Gehen wir von einer Blockverschlüsselung von n-Bit-Blöcken mit l-Bit-Schlüsseln aus. Dann haben die Tabellen die Länge 2^l, es gibt also 2^{2l} Vergleiche. Da es 2^n verschiedene mögliche Werte gibt, kann man etwa $N_1 = 2^{2l-n}$ Übereinstimmungen erwarten. (Annahmen über die Zufälligkeit der Werte implizit. Die erste Übereinstimmung ist wegen des Geburtstagsphänomens nach etwa $2^{n/2}$ Versuchen zu erwarten, aber das nützt hier kaum.)

Probiert man die gefundenen Schlüsselpaare mit einem weiteren bekannten Klartext, so bleiben etwa $N_2 = N_1/2^n = 2^{2l-2n}$ Kandidaten übrig. Nach der Prüfung von insgesamt t bekannten Klartextblöcken kann man noch $N_t = 2^{2l-tn}$ Kandidaten erwarten – aber natürlich mindestens einen, nämlich den richtigen.

Eine eindeutige Lösung wird also im allgemeinen erreicht, wenn

$$t \geq \frac{2l}{n}.$$

Beispiele

1. DES, $n = 64$, $l = 56$: $N_1 = 2^{48}$, $N_2 = 2^{-16}$. *Es werden ungefähr* 2 *bekannte Klartextblöcke benötigt.*

2. IDEA, $n = 64$, $l = 128$: $N_1 = 2^{192}$, $N_2 = 2^{128}$, $N_3 = 2^{64}$, $N_4 = 1$. *Es werden ungefähr* 4 *bekannte Klartextblöcke benötigt.*

3. AES, $n = 128$, $l = 128$: $N_1 = 2^{128}$, $N_2 = 1$. *Es werden ungefähr* 2 *bekannte Klartextblöcke benötigt.* Allerdings ist wegen $\#K = 2^{128}$ die Zahl der benötigten Speicherplätze hier sehr weit außerhalb der Möglichkeiten.

Time-Memory-Tradeoff

Eine allgemeinere Überlegung führt zu einer Ausbalancierung von Zeit und Speicherplatz („Time-Memory-Tradeoff“): Man kann bei dem Treffpunkt-Angriff Speicherplätze auf Kosten von Rechenzeit sparen, wenn man nur Teiltabellen anlegt:

Hält man in einem Durchgang jeweils s Bits von h und k fest, so benötigt man jeweils 2^{l-s} Speicherplätze für die Tabellen der $f_k(a)$ bzw. $f_h^{-1}(c)$. Zur Kompensation muss man 2^{2s} solche Durchgänge mit je einem Tabellenpaar-Abgleich machen. Der Aufwand beträgt:

$2 \cdot 2^{l-s}$	Verschlüsselungsoperationen für ein Tafelpaar,
2^{2s}	Tafelpaar-Abgleiche, also insgesamt
$2 \cdot 2^{l+s}$	Verschlüsselungsoperationen,
$2 \cdot 2^{l-s}$	Speicherplätze.

Das Produkt aus der Anzahl der Verschlüsselungsoperation und der benötigten Speicherplätze ist $4 \cdot 2^{2l}$, unabhängig von s. *Der Angreifer kann also seine Ressourcen flexibel einsetzen.*

Beispiel DES: Hat der Angreifer 128 Terabyte Speicher zur Verfügung, so kann er 2 Tabellen von je 2^{40} Blöcken anlegen, also $s = 56 - 40 = 16$ wählen. Er benötigt dann insgesamt $2 \cdot 2^{72}$ Verschlüsselungsoperationen. Das liegt für den größten Geheimdienst der Welt zweifellos im Bereich des Machbaren.

1.4 Dreifach-Chiffren

Die im vorigen Abschnitt gezeigte Schwäche der Zweifach-Chiffren führt dazu, dass man, wenn man eine Verstärkung braucht, zu Dreifach-Chiffren übergeht. Üblicherweise verwendet man das „EDE-Schema“ (Encryption, Decryption, Encryption)

$$f_g \circ f_h^{-1} \circ f_k \quad \text{für } g, h, k \in K.$$

Es hat den Vorteil, dass man mit der Schlüsselwahl $g = h = k$ die Kompatibilität mit der Einfach-Chiffre hat.

Der Treffpunkt-Angriff kann hier natürlich auch durchgeführt werden und ergibt, dass die Bitlänge für die vollständige Suche gegenüber der Einfach-Chiffre zwar nicht verdreifacht, aber doch immerhin (mehr als) verdoppelt ist.

Üblich ist auch ein vereinfachtes Schema: die Dreifach-Verschlüsselung mit zwei Schlüsseln:

$$f = f_k \circ f_h^{-1} \circ f_k \quad \text{für } h, k \in K.$$

Dieses Schema hat eine Schwäche bei einem Angriff mit gewähltem Klartext, die allerdings nur für Paranoiker bedenklich ist. Die Situation sei

$$\begin{array}{ccccccccc} \Sigma^* & \xrightarrow{f_k} & \Sigma^* & \xrightarrow{f_h^{-1}} & \Sigma^* & \xrightarrow{f_k} & \Sigma^*, \\ a & \mapsto & b & \mapsto & b' & \mapsto & c. \end{array}$$

Schritt 1: Mit $\#K$ Verschlüsselungsschritten und $\#K$ Speicherplätzen wird für jeden Zwischenwert b_0 die Tabelle

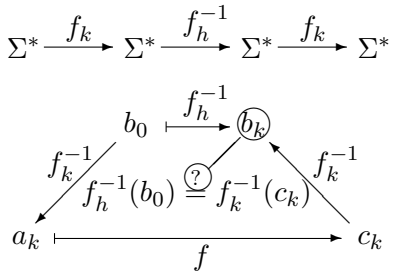
$$\{f_h^{-1}(b_0) \mid h \in K\}$$

vorausberechnet.

Schritt 2: Dann wird für alle Schlüssel $k \in K$ berechnet:

$$\begin{aligned} a_k &:= f_k^{-1}(b_0), \\ c_k &:= f(a_k), \\ b_k &:= f_k^{-1}(c_k); \end{aligned}$$

die zweite Zuweisung ist möglich, weil wir einen Angriff mit gewähltem Klartext durchführen, d. h., wir können f auf beliebige Klartexte anwenden. Das ganze benötigt $5 \cdot \#K$ einfache Verschlüsselungsschritte. Falls $b_k = f_k^{-1}(b_0)$, ist ein Kandidaten-Schlüsselpaar (h, k) gefunden, das weiter untersucht wird.



Der effizienteste bekannte Angriff wird beschrieben in:

- Van Oorschot/Wiener: A known plaintext attack on two-key triple encryption. Eurocrypt 90.

1.5 Kaskaden verschiedener Chiffren

Beispiele

1. Monoalphabetische Substitutionen und Transpositionen sind vertauschbar. Eine Komposition von mehr als jeweils einer solchen Operation bringt also nichts, da sie einzeln für sich je eine Gruppe bilden. Eine Komposition aus einer monoalphabetischen Substitution und einer Transposition ist noch ziemlich leicht sogar ohne bekannten Klartext zu brechen, da die wichtigsten Buchstaben aufgrund ihrer Häufigkeiten erkannt werden.

2. Auch für periodische polyalphabetische Chiffren und Transpositionen gilt diese Bemerkung – solange die Periodenlänge nicht nach jedem Schritt geändert wird. Hier kann man schon recht komplizierte Chiffren erzeugen.

3. Die Enigma führte die Komposition aus einer monoalphabetischen, einigen polyalphabetischen verschiedener Periode und nochmal einer monoalphabetischen Substitution aus. Das ganze war zusammengenommen auch wieder nur eine polyalphabetische Substitution sehr langer Periode.

4. Die ADFGVX-Chiffre der deutschen Wehrmacht im 1. Weltkrieg bestand aus einer Substitution gefolgt von einer Spaltentransposition; die Substitution wurde ausgeführt, indem die 26 Buchstaben und 10 Ziffern in ein 6-mal-6-Quadrat nach Vorgabe eines Schlüssels geschrieben wurden und jedes Zeichen durch seine Koordinaten in diesem Quadrat ersetzt wurde, welche mit A, D, F, G, V, X bezeichnet waren. Es gelang den Franzosen (Painvin und Givierge) zum Teil, diese Chiffre zu brechen.

5. Die Komposition von monoalphabetischer Chiffre und einer Autokey-Chiffre werden wir bald als „Betriebsmodus bei Blockverschlüsselung“ kennen lernen. Die Autokey-Chiffre erschwert einige Angriffe, erhöht aber insgesamt die Sicherheit nur unwesentlich.

6. Ferner sei noch einmal daran erinnert, dass die Drehscheiben-Chiffre nach Porta sich als Komposition einer monoalphabetischen Substitution und einer Belaso-Chiffre darstellen ließ – sogar in beiden möglichen Reihenfolgen.

Insgesamt kann man sagen, dass Kaskaden verschiedener Chiffren oft zu Verbesserungen der Sicherheit führen, aber durchaus nicht immer. Es ist in jedem Fall eine sorgfältige Analyse der entstehenden Produkt-Chiffre nötig, bevor man ihr vertraut.