DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Grundlagen der Kryptographie und Steganographie

Studienarbeit

für die Prüfung zum

Bachelor of Science

des Studiengangs Informatik / Angewande Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

Jens Döllmann

Abgabedatum 17. Mai 2021

Bearbeitungszeitraum	2 Semester
Matrikelnummer	8876462
Kurs	TINF18B4
Gutachter der Studienakademie	Ralf Brune

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Symmetrische Verschlüsselung 3

1.2 Modulare Arithmetik 4

1.3 Die Verschiebe- oder Cäsar-Chiffre 8

2 Stromchiffren **10**

2.1 Stromchiffren und Blockchiffren 10

2.2 Die Ver- und Entschlüsselung mit Stromchiffren 12

2.3 Zufallszahlengeneratoren 15

2.4 Das One-Time-Pad und praktische Stromchiffren 16

Tabellenverzeichnis

1.1 Enkodierung des lateinischen Alphabets 8

2.1 Wahrheitstabelle der Addition modulo 2 13
2.2 Wahrheitstabelle der Exklusiv-Oder-Verknüpfung 13

Abbildungsverzeichnis

1.1 Die Kryptologie und ihre Untergebiete (Paar und Pelzl 2010, S. 3) 2
1.2 Kommunikation über einen unsicheren Kanal 3
1.3 Kommunikation mit symmetrischer Verschlüsselung 4

2.1 Unterteilung der Symmetrischen Chiffren (Paar und Pelzl 2010, S. 29) . . . 10
2.2 Das Prinzip der Verschlüsselung von n Bits mittels Strom- (a) und Blockchiffre (b) (Paar und Pelzl 2010, S. 30) 11
2.3 Der Ver- und Entschlüsselungsprozess bei Stromchiffren 14
2.4 Praktische Stromchiffre mit Schlüsselstromerzeugung (Paar und Pelzl 2010, S. 38) 17

Glossar

CSPRNG *cryptographically secure PRNG*. 16, 18

OTP One-Time-Pad. 16, 17, 18

PRNG *pseudo random number generator*. IV, 15, 16, 18

RNG *random number generator*. 15

TRNG *true random number generator*. 15, 16, 17

Kapitel 1

Einleitung

Redet man heutzutage über das Thema Kryptographie, sind im Gespräch wahrscheinlich Themen wie E-Mail-Verschlüsselung, Internetprotokolle oder Anwendungen im Bankenwesen. Auch bekannt sind die Angriffe auf kryptographische Systeme, wie zum Beispiel die Entzifferung der durch die Enigma-Chiffriermaschine verschlüsselten deutschen Funksprüche während des Zweiten Weltkrieges. Es scheint als wäre Kryptographie stark mit den modernen elektronischen Kommunikationstechniken verbunden. Dies ist allerdings nicht so: Frühe Formen der Kryptographie gehen zurück bis etwa 2000 v. Chr., als bereits im antiken Ägypten neben den Standard-Hieroglyphen zusätzlich auch „geheime“ Zeichen verwendet wurden (Paar und Pelzl 2010, S. 2). Es werden prinzipiell zwei unterschiedliche kryptographische Verfahren unterschieden, diese sind Symmetrische- und Asymmetrische Algorithmen. Die symmetrische Verschlüsselung ist seit langer Zeit ein fester Bestandteil der Kryptographie, mit bekannten historischen Verfahren wie die Cäsar-Chiffre welche bereits im antiken Rom für das Verschlüsseln von Nachrichten verwendet wurde. Asymmetrische Verschlüsselung hingegen ist eine gänzlich neue Form der Kryptographie, Whitfield Diffie, Martin Hellman und Ralph Merkle haben die Idee im Jahr 1976 erstmalig öffentlich eingeführt (ebd., S. 3). Eine Übersicht über das Gebiet der Kryptographie ist in Abbildung 1.1 zu sehen. Es ist zu bemerken, dass an oberster Stelle nicht die Kryptographie, sondern der Oberbegriff Kryptologie zu finden ist, welche sich in die zwei großen Bereiche unterteilt:

Kryptographie Die Wissenschaft eine Nachricht so zu verändern, dass ihr Sinn nur von dem Empfänger verstanden werden kann, für den sie bestimmt ist.

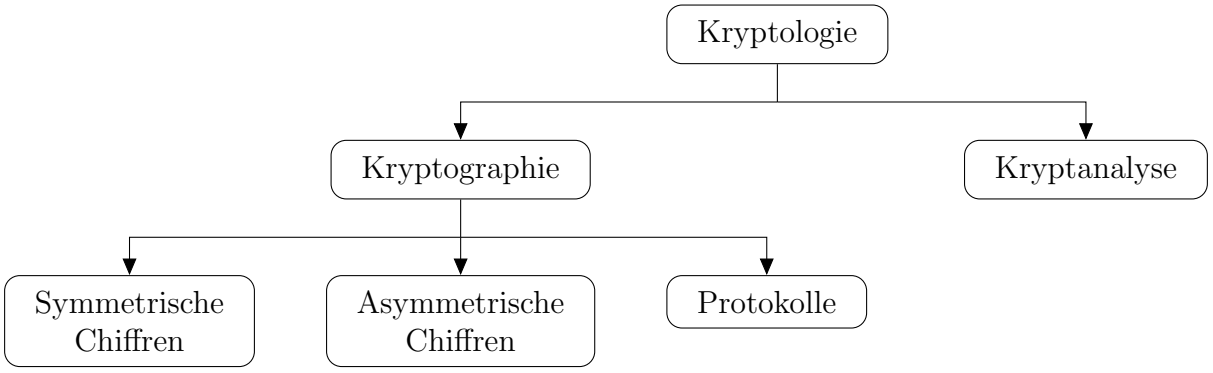

Abbildung 1.1: Die Kryptologie und ihre Untergebiete (Paar und Pelzl 2010, S. 3)

Kryptanalyse Die Wissenschaft ein kryptographisches System zu analysieren mit dem Ziel mögliche Schwachstellen aufzudecken. Die Kryptanalyse ist ein äußerst wichtiger Teil der Kryptologie. Ohne Personen welche versuchen ein kryptographisches System zu brechen, wird man nie herausfinden können ob das System wirklich sicher ist. Ein starkes Kryptoverfahren sollte dem *Kerckhoffs's principle* unterliegen, welches im Jahr 1883 von Auguste Kerckhoffs postuliert wurde und von Paar und Pelzl durch folgende Definition beschrieben ist (2010, S. 11):

Definition 1.0.1 (*Kerckhoffs's principle*)**.** „*A cryptosystem should be secure even if the attacker knows all details about the system, with the exception of the secret key. In particular, the system should be secure when the attacker knows the encryption and decryption algorithms.*“

Auf den ersten Blick scheint das *Kerckhoffs's principle* nicht sonderlich intuitiv. Es sei einfach zu glauben, dass ein System sicherer sein muss, wenn die Details der Implementierung geheim gehalten werden. In der Regel ist dies aber nicht so. Ein Kryptoverfahren bleibt nicht für immer geheim und die Vergangenheit hat gezeigt, dass ein System dessen geheimes Design an die Öffentlichkeit gelangt, fast immer unsicher ist. Ein hierfür gutes Beispiel ist das Content Scrambling System (CSS) für das Verschlüsseln von DVD-Videoinhalten. Trotz großer Bemühungen der Industrie die Funktionsweise von CSS geheim zu halten, gelang das Design durch Reverse Code Engineering schnell an die Öffentlichkeit. Es zeigten sich Mängel in der Implementierung, welche das Brechen der Verschlüsselung mit sehr geringen Aufwand ermöglichten (Barry 2004).

1.1 Symmetrische Verschlüsselung

Denkt man an die Teilbereiche der Kryptographie, ist die Symmetrische Verschlüsselung das wohl klassischste Beispiel. Zwei Parteien kommunizieren mit einem Algorithmus zum Ver- und Entschlüsseln von Nachrichten und haben sich auf einen gemeinsamen geheimen Schlüssel geeinigt. Wie es in der Literatur sehr beliebt ist, wird die Idee der symmetrischen Verschlüsselung mit einem einfachen Beispiel eingeführt (Paar und Pelzl 2010, S. 4–6): Zwei Parteien Alice und Bob möchten über einen unsicheren Kanal Nachrichten untereinander austauschen. Ein unsicherer Kanal ist hierbei lediglich die Kommunikationsstrecke, z.B. das Internet, die Luftschnittstelle bei WLAN und Mobilfunk oder jedes andere Medium, über das sich digitale Daten übertragen lassen.

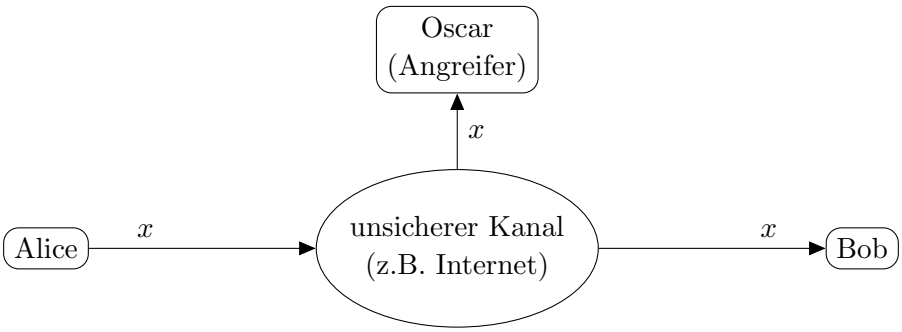

Abbildung 1.2: Kommunikation über einen unsicheren Kanal

Es ist klar warum Alice und Bob gerne geheime Nachrichten austauschen würden. Alice möchte sich an ihrem Bankkonto anmelden und sendet ihr Passwort zu Bob. Ein potenzieller Angreifer Oscar soll die Passwörter von Alice nicht in Klartext mitlesen können. In einer solchen Situation bietet die Symmetrische Verschlüsselung eine gute Lösung: Bevor Alice ihr Passwort sendet, verschlüsselt sie es mit einem symmetrischen Algorithmus. Bob invertiert die Verschlüsselung und erhält die unverschlüsselte Nachricht. Wurde für die Verschlüsselung ein sicherer Algorithmus gewählt, erscheint die Nachricht für Oscar nur wie eine zufällige Folge von Bits.

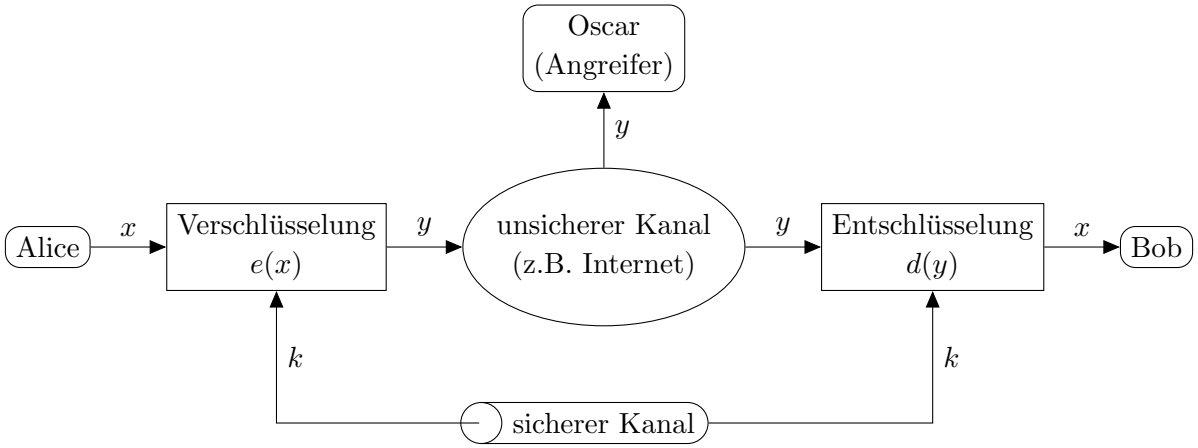

Abbildung 1.3: Kommunikation mit symmetrischer Verschlüsselung

Die Variablen x, y und k aus Abbildung 1.3 haben in der Kryptographie eine besondere Bedeutung:

- x ist der Klartext (engl. *plaintext*).
- y ist das Chiffrat oder der Geheimtext (engl. *ciphertext*).
- k ist der Schlüssel (engl. *key*).
- $e(\cdot)$ ist die Verschlüsselung (engl. *encryption*).
- $d(\cdot)$ ist die Entschlüsselung (engl. *decryption*), d.h. die Umkehrfunktion von e.

Für die Symmetrische Verschlüsselung wird der geheime Schlüssel k benötigt. Dieser muss vor der Kommunikation auf einem sicheren Weg zwischen Alice und Bob verteilt werden.

1.2 Modulare Arithmetik

Fast alle kryptographischen Algorithmen, sowohl Symmetrische als auch Asymmetrische Chiffren, basieren auf Arithmetik in einer endlichen Menge von ganzen Zahlen (Paar und Pelzl 2010, S. 13). Dies steht im Gegensatz zu der Mathematik (und dem Alltagsleben) in der wir es gewöhnt sind in unendlichen Mengen zu rechnen, z.B. die natürlichen Zahlen oder die reellen Zahlen. Die modulare Arithmetik, d.h. die Division mit Rest, bietet eine gute Möglichkeit um in diesen begrenzten Mengen rechnen zu können.

Lemma 1.2.1 (Remmert und Ullrich 2008, S. 179–180)**.** *Folgende Aussagen über drei ganze Zahlen a, b, m, wobei $m > 0$, sind äquivalent:*

i) a und b lassen bei Division mit Rest durch m denselben Rest.

ii) Die Differenz $a - b$ ist durch m teilbar.

Beweis. $\exists\, q_1, q_2, r_1, r_2 \in \mathbb{Z}$
Es seien $a = q_1 m + r_1$ und $b = q_2 m + r_2$ mit $0 \leq r_1, r_2 < m$, die Gleichungen, die bei Division mit Rest entstehen.
i) $\Rightarrow$ ii):
Es gilt: $r_1 = r_2$.
Zu zeigen: $m|(a - b)$.

$$\begin{aligned} & a - b = q_1 m + r_1 - q_2 m + r_2 \\ & a - b = (q_1 - q_2)m + r_1 - r_2 \\ \iff\ & a - b = (q_1 - q_2)m \end{aligned}$$

Aus der letzten Gleichung folgt: $m|(a - b)$.
ii) $\Rightarrow$ i):
Es gilt: $m|(a - b)$.
Zu zeigen: $r_1 = r_2$.

$$\begin{aligned} & m|(a - b) \\ \Leftrightarrow\ & m|(q_1 m + r_1) - (q_2 m + r_2) \\ \Leftrightarrow\ & m|(q_1 - q_2)m + (r_1 - r_2) \end{aligned}$$

$m|(q_1 - q_2)m + (r_1 - r_2)$ und $m|(q_1 - q_2)m$, weshalb gelten muss: $m|(r_1 - r_2)$. [1]
Entweder $r_1 - r_2$ ist ein Vielfaches von m oder 0. Da sich r_1 und r_2 im Bereich $[0, m)$ befinden ist die einzige Lösung $r_1 - r_2 = 0$. Es folgt: $r_1 = r_2$. □

[1] Aus $a|b$ und $a|c$ folgt $a|(xb + yc)$, $\forall\, x, y \in \mathbb{Z}$ (Remmert und Ullrich 2008, S. 23).
Also mit konkreten Werten $m|(q_1 - q_2)m + (r_1 - r_2) - (q_1 - q_2)m \Leftrightarrow m|(r_1 - r_2)$.

Nach Gauß nennt nennt man zwei Zahlen $a, b \in \mathbb{Z}$, die bei der Division durch m denselben Rest ergeben, *kongruent modulo m*. Anstelle der schwerfälligen Teilbarkeitsschreibweise $m|(a-b)$ führte Gauß folgende Schreibweise ein (Remmert und Ullrich 2008, S. 180):

$$a \equiv b \mod m \quad \text{oder kürzer:} \quad a \equiv b \quad (m) \tag{1.1}$$

*B*eispiel 1.2.1. Es sind $a = 29$ und $m = 8$. Man schreibt

$$29 \equiv 5 \mod 8 \quad \text{oder als Gleichung:} \quad 29 = 3 \cdot 8 + 5$$

und deshalb $8|(29-5)$. △

Die Gleichung der Modulo Rechnung hat unendlich viele Lösungen. Zu einem gegeben m gibt es beliebig viele Zahlen a, welche den selben Rest ergeben.

*B*eispiel 1.2.2. Die folgenden Zahlen erfüllen die Gleichung $a \mod 8 = 5$. Sie werden kongruent modulo 8 genannt:

$$29 \equiv 21 \equiv 13 \equiv 5 \equiv -3 \equiv -11 \mod 8$$

Elemente welche bei der Modulo Rechnung den gleichen Rest ergeben, können in eine Klasse zusammengefasst werden, diese nennt man dann eine Restklasse bezüglich m. Die acht Restklassen bezüglich 8 sind:

$$\begin{gathered}\{\ldots, -24, -16, -8, 0, 8, 16, 24, \ldots\} \\ \{\ldots, -25, -17, -9, 1, 9, 17, 25, \ldots\} \\ \vdots \\ \{\ldots, -17, -9, -1, 7, 15, 23, 31, \ldots\}\end{gathered}$$

△

Alle Elemente einer Restklasse verhalten sich gleich. Zu einem gegebenen Modulo m spielt es keine Rolle, welches Element der Restklasse für eine Berechnung ausgewählt wird. Diese Eigenschaft ist von großem Nutzen, vor allem bei Berechnungen mit großen Zahlen, wie es in der Kryptographie oft der Fall ist.

*B*eispiel 1.2.3 (Paar und Pelzl 2010, S. 15–16). Die Hauptoperation in vielen Asymmetrischen Chiffren ist die Exponentiation der Form $x^e \mod m$, wobei x, e, m sehr große ganze Zahlen sind. Anhand eines Beispiels können zwei Formen der modularen Exponentiation gezeigt werden. Es soll das Ergebnis der Berechnung $3^8 \mod 7$ ermittelt werden. Im ersten Beispiel wird das Ergebnis einfach ausgerechnet, und im zweiten Beispiel wird zwischen den Restklassen gewechselt:

1. $3^8 = 6561 \equiv 2 \mod 7$, weil $6561 = 937 \cdot 7 + 2$
   Wir erhalten das relative große Zwischenergebnis 6561 obwohl wir wissen, dass das Ergebnis im Bereich $[0, 6]$ liegen muss.
2. Es ist $3^3 = 27 \equiv -1 \mod 7$. Man schreibt:

$$3^8 = (3^3)^2 \cdot 3^2 \equiv (-1)^2 \cdot 9 \equiv 2 \mod 7$$

   Indem man das Zwischenergebnis $3^3 = 27$ mit einem kleineren Element aus der selben Restklasse ersetzt, kann das Ergebnis effizient ermittelt werden. Zwischenergebnisse werden nie größer als 27 und man könnte die Berechnung mit wenig Aufwand auch ohne Taschenrechner durchführen.

△

Bemerkung 1.2.1. Paar und Pelzl (ebd., S. 16) haben sich in ihrem Buch darauf geeinigt, in der Kongruenzrelation (1.1) ein b für gewöhnlich so zu wählen, dass:

$$0 \leq b < m$$

Man schreibt somit $27 \equiv 6 \ (7)$ und nicht $27 \equiv -1 \ (7)$ oder $27 \equiv 13 \ (7)$. [2] Mathematisch macht es jedoch keinen Unterschied.

Um das bekannte Verschlüsselungsverfahren, die Cäsar-Chiffre, im nächsten Abschnitt besser beschreiben zu können, wird an dieser Stelle der Ring $\mathbb{Z}_m$ definiert:

[2] Mit dieser Vereinbarung ist das b der Relation $a \equiv b \ (m)$ eine Lösung für die Gleichung $a \mod m = r = b$.

Definition 1.2.1 (Der Ring $\mathbb{Z}_m$ der Reste modulo m)**.**

1. Der Ring $\mathbb{Z}_m$ mit $m \in \mathbb{N}$ und $m > 1$ ist definiert als die Menge $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$
2. Zu je zwei Elementen $a, b \in \mathbb{Z}_m$ existiert eindeutig in $\mathbb{Z}_m$
   (a) Eine Summe $a + b \mod m$
   (b) Ein Produkt $a \cdot b \mod m$

1.3 Die Verschiebe- oder Cäsar-Chiffre

Die Cäsar-Chiffre ist das vielleicht bekannteste historische Verschlüsselungsverfahren, es wird von Paar und Pelzl auf Seiten 18-19 eingeführt (2010). Bei der Cäsar-Chiffre handelt es sich um eine spezielle Form der Buchstabensubstitution. Jedes Zeichen im Klartext wird zur Verschlüsselung um einen bestimmten Wert im Alphabet verschoben. Um die Cäsar-Chiffre mathematisch zu beschreiben, muss das zu Grunde liegende Alphabet enkodiert werden. Eine Möglichkeit ist die fortlaufende Nummerierung der Buchstaben. Das so kodierte lateinische Alphabet ist in Tabelle 1.1 zu sehen.

Tabelle 1.1: Enkodierung des lateinischen Alphabets

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Würde ein Buchstabe zu weit verschoben werden, wird von vorne, also bei dem ersten Buchstabe weitergemacht. Sowohl Klartext- als auch Geheimtextbuchstaben sind somit Teil des Rings $\mathbb{Z}_{26}$. Die Ver- und Entschlüsselung kann nun folgendermaßen ausgedrückt werden:

Definition 1.3.1 (Cäsar-Chiffre)**.** Es seien $x, y, k \in \mathbb{Z}_{26}$. Dann gilt:

Verschlüsselung: $y = e_k(x) \equiv x + k \mod 26$

Entschlüsselung: $x = d_k(y) \equiv y - k \mod 26$

Das Verschlüsselungsverfahren funktioniert.

Beweis. Die Entschlüsselung der Verschlüsselung ergibt den Klartext.
Zu zeigen: $d_k(e_k(x)) = x$.

$$d_k(e_k(x)) = x \equiv x + k - k \equiv x \mod 26$$

□

*B*eispiel 1.3.1. Es sei $k = 9$ und der Klartext:

$$\texttt{ATTACK} = x_1, x_2, \ldots, x_6 = 0, 19, 19, 0, 2, 10$$

Der Geheimtext wird wie folgt berechnet:

$$\begin{aligned} e_9(0) &\equiv 9 \mod 26 \\ e_9(2) &\equiv 11 \mod 26 \\ e_9(10) &\equiv 19 \mod 26 \\ e_9(19) &\equiv 28 \equiv 2 \mod 26 \end{aligned}$$

$$y_1, y_2, \ldots, y_6 = 9, 2, 2, 9, 11, 19 = \texttt{JCCJLT}$$

△

Wie zu erwarten ist die Cäsar-Chiffre natürlich nicht besonders sicher. Es gibt nur 26 verschiedene Schlüssel (wobei $k = 0$ den Klartext nicht verändert), welche schnell alle ausprobiert werden können. Zusätzlich haben Klartext und Geheimtext die selben statistischen Eigenschaften. Klartextbuchstaben werden immer auf die selben Geheimtextbuchstaben abgebildet. Dies erlaubt es eine Häufigkeitsanalyse der Buchstaben durchzuführen.

Kapitel 2

Stromchiffren

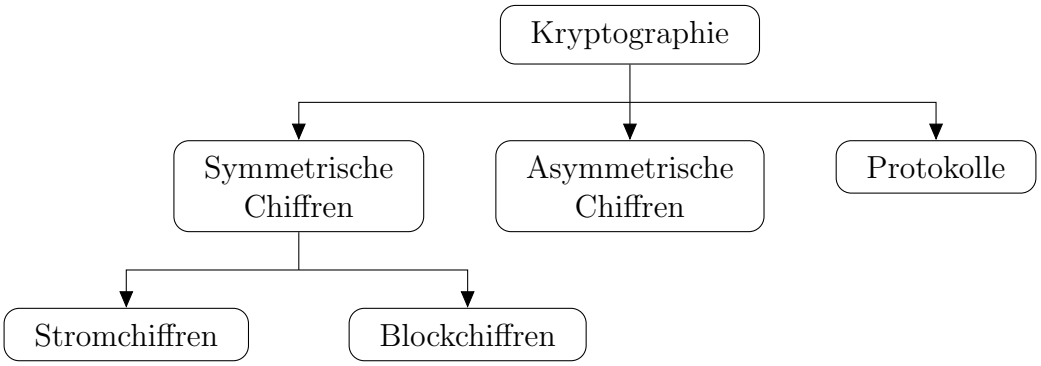

Abbildung 2.1: Unterteilung der Symmetrischen Chiffren (Paar und Pelzl 2010, S. 29)

Werfen wir in Abbildung 2.1 einen genaueren Blick auf die Algorithmen der Kryptographie, stellen wir fest: Das Gebiet der symmetrischen Verschlüsselungsverfahren kann unterteilt werden in Strom- und Blockchiffren. In diesem Kapitel soll die Funktionsweise von Stromchiffren untersucht und der Unterschied zu den Blockchiffren erläutert werden. Außerdem wird gezeigt welche Rolle hierbei die Zufallszahlengeneratoren spielen.

2.1 Stromchiffren und Blockchiffren

Die symmetrischen Verschlüsselungsverfahren sind unterteilt in Strom- und Blockchiffren. Während beide Verfahren das Ziel haben Informationen zu verschlüsseln, ist die jeweilige Methode eine unterschiedliche. Stromchiffren verschlüsseln jedes Bit im Klartext einzeln,

während Blockchiffren pro Durchlauf mehrere Bits verschlüsseln können. Abbildung 2.2 zeigt diesen prinzipiellen Unterschied, für den Fall, dass n Bits verschlüsselt werden sollen.

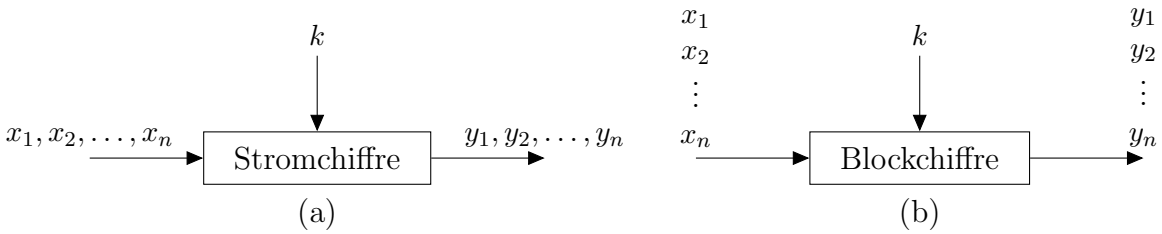

Abbildung 2.2: Das Prinzip der Verschlüsselung von n Bits mittels Strom- (a) und Blockchiffre (b) (Paar und Pelzl 2010, S. 30)

Blockchiffren arbeiten mit Datenblöcken fester Länge, wobei ein Großteil der relevanten Algorithmen eine Eingangsweite von 64 oder 128 Bits besitzen. Prominente Beispiele sind Verfahren wie der Data Encryption Standard (DES, Blocklänge 64 Bit, Paar und Pelzl 2010, S. 55–58) oder der Advanced Encryption Standard (AES, Blocklänge 128 Bit, ebd., S. 87–90). Generell lassen sich die folgenden drei Punkte zusammenfassen (ebd., S. 31):

1. Blockchiffren werden in der Praxis, vor allem für die Verschlüsselung im Internet, häufiger eingesetzt als Stromchiffren.
2. Stromchiffren sind oft klein und schnell und deshalb attraktiv für Anwendungen, bei denen vergleichsweise wenig Rechenleistung zur Verfügung steht, beispielsweise bei Mobilgeräten oder anderen eingebetteten Systemen. Ein bekanntes Verfahren ist der A5/1 Algorithmus, welcher Teil des GSM-Mobilfunkstandards ist und die Gesprächsdaten an der Luftschnittstelle verschlüsselt. A5/1 gilt aufgrund der kurzen Schlüssellänge von 64 Bit nicht mehr als uneingeschränkt sicher, erschwert aber dennoch das einfache Abhören.
3. In der Vergangenheit galt der generelle Gedanke, dass Stromchiffren effizienter seien als Blockchiffren. Im Allgemeinen gilt diese Annahme heutzutage jedoch nicht mehr. Moderne Verfahren wie AES können sowohl in Hardware als auch Software sehr effizient implementiert werden.

2.2 Die Ver- und Entschlüsselung mit Stromchiffren

Wie oben erwähnt, verschlüsseln Stromchiffren jedes Bit im Klartext einzeln. Hierfür wird durch einen Schlüssel ein geheimer Bitstrom errechnet, welcher paarweise mit den Bits des Klartextes kombiniert wird. Die Ver- und Entschlüsselung ist verblüffend einfach, es handelt sich in beide Richtungen um eine einfache Addition im Ring $\mathbb{Z}_2$.

Definition 2.2.1 (Ver- und Entschlüsselung mit Stromchiffren, Paar und Pelzl 2010, S. 31)**.** Es seien $x_i, y_i, s_i \in \{0, 1\}$ die einzelnen Bits aus Klartext, Geheimtext und Schlüsselstrom. Es gilt:

Verschlüsselung: $y_i = e_{s_i}(x_i) \equiv x_i + s_i \mod 2$

Entschlüsselung: $x_i = d_{s_i}(y_i) \equiv y_i + s_i \mod 2$

Betrachtet man die Ver- und Entschlüsselungsfunktion, fallen drei Aspekte auf welche besprochen werden müssen (ebd., S. 31–34):

1. Warum ist Verschlüsselung und Entschlüsselung die selbe Funktion?
2. Warum ist einfache Addition im Ring $\mathbb{Z}_2$ eine gute Verschlüsselung?
3. Was sind die Eigenschaften der Schlüsselstrombits s_i?

Warum ist Verschlüsselung und Entschlüsselung die selbe Funktion?
Bis auf spezielle Kürzungsregeln erlauben Kongruenzen (ganz analog wie mit Gleichungen) das Ausführen der elementaren Rechenoperationen (Remmert und Ullrich 2008, S. 181–183). Mit diesem Wissen kann die Verschlüsselungsfunktion durch einfaches Umformen in die Entschlüsselungsfunktion überführt werden.

Beweis.

$$\begin{aligned} y_i &\equiv x_i + s_i \mod 2 \\ -x_i &\equiv -y_i + s_i \mod 2 \\ x_i &\equiv y_i + s_i \mod 2 \end{aligned}$$

□

Im letzten Schritt wird erneut von dem Wechsel innerhalb der Restklasse Gebrauch gemacht, denn es gilt $-1 \equiv 1 \mod 2$.

Warum ist einfache Addition im Ring $\mathbb{Z}_2$ eine gute Verschlüsselung?
Das Rechnen in $\mathbb{Z}_2$ liefert aufgrund der Division mit Rest nur Ergebnisse in der Menge $\{0, 1\}$. Dieses Verhalten ist sehr hilfreich, denn es ermöglicht das Ausdrücken der Arithmetik durch einfache boolesche Algebra. Betrachtet man die Wahrheitstabelle 2.1 der Addition in $\mathbb{Z}_2$, kann sofort eine weitere Beobachtung gemacht werden: Die Addition modulo 2 ist äquivalent zu der Exklusiv-Oder-Verknüpfung durch ein XOR-Gatter.

Tabelle 2.1: Wahrheitstabelle der Addition modulo 2

x_i	s_i	$y_i \equiv x_i + s_i \mod 2$
0	0	0
0	1	1
1	0	1
1	1	0

Das XOR-Gatter spielt eine wesentliche Rolle in vielen kryptographischen Verfahren. Es besitzt besondere Eigenschafen, welche es von anderen Logikgattern unterscheidet, diese sollen jetzt untersucht werden. Angenommen es soll das Klartextbit $x_i = 0$ verschlüsselt werden. In der Wahrheitstabelle des XOR-Gatters befindet man sich demnach in der ersten oder zweiten Zeile:

Tabelle 2.2: Wahrheitstabelle der Exklusiv-Oder-Verknüpfung

x_i	s_i	$y_i = x_i \oplus s_i$
0	**0**	**0**
0	**1**	**1**
1	0	1
1	1	0

Je nach Schlüsselbit s_i ist das Geheimtextbit y_i gegeben als $y_i = 0 \oplus 0 \vee 0 \oplus 1 = 0 \vee 1$. Verhalten sich die Schlüsselbit unvorhersehbar , d.h. sie sind mit genau 50 prozentiger Wahrscheinlichkeit entweder null oder eins, ist es nur durch das Chiffrat nicht möglich auf den Klartext zu schließen. Zu jedem Zeitpunkt hat ein Angreifer nur eine 50 prozentige Chance den richtigen Klartext zu erraten. Gleichermaßen kann argumentiert werden, sei $x_i = 1$. Diese Symmetrie unterscheidet das XOR-Gatter von anderen Logikgattern, zusätzlich ist es die einzige Verknüpfung welche durch doppeltes Anwenden invertierbar ist. Abbildung 2.3 zeigt den Ver- und Entschlüsselungsprozess bei Stromchiffren.

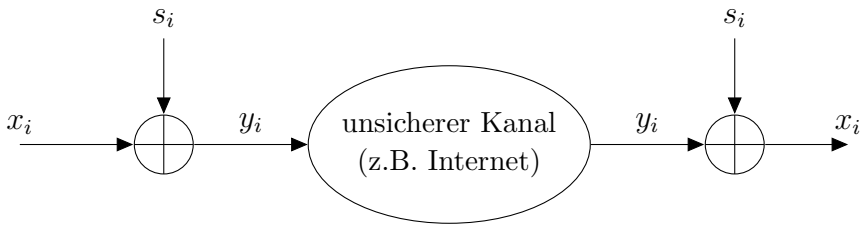

Abbildung 2.3: Der Ver- und Entschlüsselungsprozess bei Stromchiffren

In einem Beispiel soll jetzt ein einfacher Informationsaustausch demonstriert werden:

*B*eispiel 2.2.1. Alice möchte Bob eine Nachricht senden. Sie verschickt den Buchstaben `P`, wobei dieser zur Verschlüsselung in ASCII-Zeichenkodierung angegeben ist `P` $= 80_{10} = 01010000_2$. Der Schlüsselstrom wird außerdem angegeben, es seien $(s_0, s_1, \ldots, s_7) = 00111010_2$.

$$
\begin{aligned}
& x_0, \ldots, x_7 = 01010000_2 = 80_{10} = \texttt{P} \\
& \qquad\qquad \oplus \\
& s_0, \ldots, s_7 = 00111010_2 \qquad\qquad \textbf{Alice} \\
& y_0, \ldots, y_7 = 01101010_2 = 106_{10} = \texttt{j} \\
& \\
& \qquad \texttt{j} = 01101010_2 \qquad\qquad \textbf{Oscar} \\
& \qquad\qquad \longrightarrow \\
& \\
& y_0, \ldots, y_7 = 01101010_2 = 106_{10} = \texttt{j} \\
& \qquad\qquad \oplus \\
& s_0, \ldots, s_7 = 00111010_2 \qquad\qquad \textbf{Bob} \\
& x_0, \ldots, x_7 = 01010000_2 = 80_{10} = \texttt{P}
\end{aligned}
$$

Vor der Übertragung wird der Großbuchstabe `P` durch die XOR-Verknüpfung in den Kleinbuchstaben `j` umgewandelt. Der Gegenspieler Oscar kann mit dieser Information aufgrund der oben beschriebenen Eigenschaften nur wenig anfangen. △

Stromchiffren erscheinen fast zu gut, um wahr zu sein, doch wie gewohnt sind die Dinge in der Realität nicht immer so einfach. Es bleibt die Letzte der drei Fragestellungen zu beantworten.

Was sind die Eigenschaften der Schlüsselstrombits?
Die Sicherheit von Stromchiffren hängt vollständig von der Qualität des Schlüsselstroms ab. Das Generieren dieser Bitfolge $(s_1, s_2, \ldots, s_n)$ bildet die zentrale Fragestellung der Verschlüsselungsmethode. Es wurde gezeigt, dass der Schlüsselstrom wie eine zufällige Bitsequenz aussehen muss. Im nächsten Abschnitt soll deshalb das Thema der Zufallszahlen diskutiert werden.

2.3 Zufallszahlengeneratoren

Wie oben beschrieben, ist die Unvorhersehbarkeit des Schlüsselstroms die wesentliche Eigenschaft von Stromchiffren. Im Folgenden sollen deshalb einige Formen von Zufallszahlengeneratoren (engl. *random number generators* (RNGs)) vorgestellt werden (Paar und Pelzl 2010, S. 35–36) (Haahr 2021).

Echte Zufallszahlengeneratoren Echte Zufallszahlengeneratoren (engl. *true random number generators* (TRNGs)) erzeugen Zahlen, welche in keiner Weise vorhersehbar und reproduzierbar sind. Notiert man beispielsweise das Ergebnis von 100 Münzwürfen und erzeugt eine Folge von 100 Bit, ist es quasi unmöglich, die selbe Sequenz ein zweites Mal zu erzeugen. Die Wahrscheinlichkeit das dies passieren würde, beträgt $1/2^{100}$, was verschwindend gering ist. Echte Zufallszahlen basieren auf physikalischen Prozessen. Beispiele umfassen Münzwurf, Würfeln, radioaktiver Zerfall oder atmosphärisches Rauschen. TRNGs werden in der Kryptographie und der Schlüsselerzeugung häufig eingesetzt.

Pseudozufallszahlengeneratoren Pseudozufallszahlengeneratoren (engl. *pseudo random number generators* (PRNGs)) generieren Zahlenfolgen basierend auf einem Startwert, welcher im Englischen oft als „*seed*“ bezeichnet wird. Die Zahlen eines PRNG sind nicht in der Art zufällig, wie man es erwarten könnte. Obwohl sie aussehen wie eine wirklich zufällige Folge, werden sie durch mathematische Formeln errechnet und sind deterministisch, d.h. die selbe Sequenz kann zu einem späteren Zeitpunkt reproduziert werden. In der Kryptographie können PRNGs aufgrund dieser Eigenschaf nicht ohne weiteres eingesetzt werden. Dennoch haben sie außerhalb der Kryptographie weitreichende Anwendungsgebiete, beispielsweise in der Simulation oder während dem Testen von Software und Hardware. Determinismus ist in diesen Bereichen oftmals eine gewünschte Eigenschaft.

Kryptographisch sichere Pseudozufallszahlengeneratoren Kryptographisch sichere Pseudozufallszahlengeneratoren (engl. *cryptographically secure PRNG*s (CSPRNGs)) sind PRNGs mit einer speziellen zusätzlichen Eigenschaft: CSPRNGs sind unvorhersehbar. Grob gesprochen bedeutet dies, dass es rechentechnisch nicht möglich ist, aus n gegeben Schlüsselstrombits $s_1, s_2, \ldots, s_n$, die folgenden Bits $s_{n+1}, s_{n+2}, \ldots, s_{n+m}$ zu berechnen. Außerdem soll es zeitlich unmöglich sein, eines der vorherigen Bits $s_{n-1}, s_{n-2}, \ldots, s_{n-m}$ zu berechnen.

2.4 Das One-Time-Pad und praktische Stromchiffren

Mit den bis hierher eingeführten Ideen zu Stromchiffren und den verschiedenen Zufallszahlengeneratoren sind alle Teile vorhanden, um ein beweisbar sicheres Verschlüsselungsverfahren zu bauen. Ein System heißt beweisbar sicher, wenn es trotz unendlich vorhandener Rechenleistung nachweislich nicht gebrochen werden kann. Paar und Pelzl haben den Begriff mit folgender Definition beschrieben (2010, S. 36):

Definition 2.4.1 (*Unconditional Security*)**.** „*A cryptosystem is unconditionally or information-theoretically secure if it cannot be broken even with infinite computational resources.*“

Das One-Time-Pad (OTP) ist ein solches Verschlüsselungsverfahren, welches dieses Kriterium erfüllt. Es ist folgendermaßen definiert (ebd., S. 37):

Definition 2.4.2 (One-Time-Pad)**.** Eine Stromchiffre heißt One-Time-Pad, wenn die folgenden Kriterien eingehalten werden:

1. Der Schüsselstrom $s_1, s_2, \ldots, s_n$ wird durch einen TRNG generiert.
2. Nur vertrauenswürdige Gesprächspartner kennen den Schüsselstrom.
3. Jedes Schlüsselstrombit s_i wird nur einmal verwendet.

Das One-Time-Pad ist beweisbar sicher.

Zu jedem Bit im Geheimtext gibt es eine Gleichung der Form:

$$\begin{aligned} y_0 &\equiv x_0 + s_0 \mod 2 \\ y_1 &\equiv x_1 + s_1 \mod 2 \\ &\vdots \end{aligned}$$

Stammt der Schlüsselstrom von einem TRNG können diese Gleichungen nicht gelöst werden. Für jedes Geheimtextbit y_i gibt es genau zwei Lösungen mit gleich großer Wahrscheinlichkeit:

$$\begin{aligned} y = 0 \equiv 1_x + 1_s \equiv 0_x + 0_s \mod 2 \\ y = 1 \equiv 1_x + 0_s \equiv 0_x + 1_s \mod 2 \end{aligned}$$

In der Praxis hat das Verschlüsseln mit OTP natürlich einen Hacken, denn warum sonst sollten heutzutage andere Verfahren eingesetzt werden?

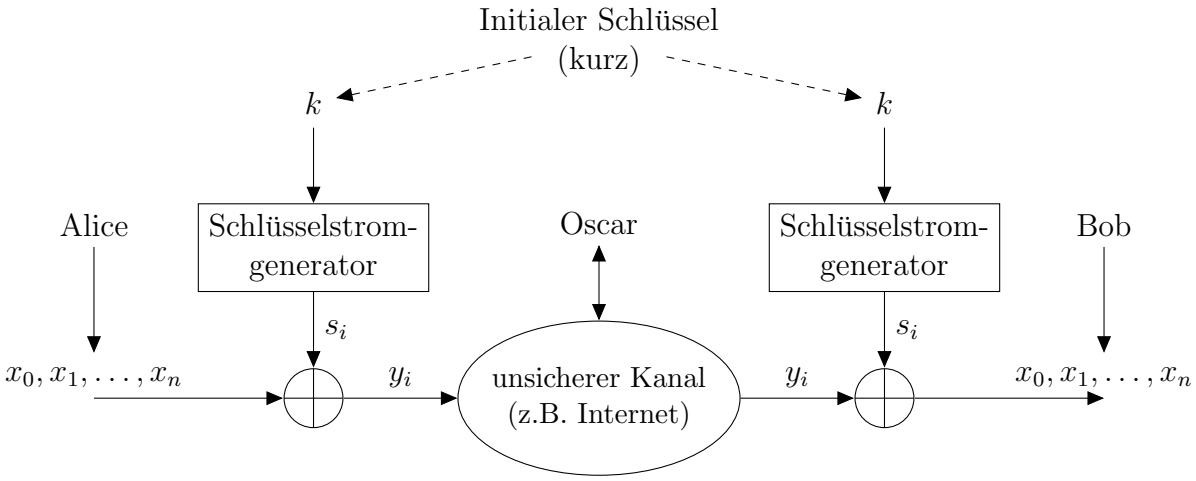

Abbildung 2.4: Praktische Stromchiffre mit Schlüsselstromerzeugung (Paar und Pelzl 2010, S. 38)

Als erster Punkt ist die Anforderung eines TRNG, dieser benötigt spezielle Hardware und ist in der Regel nicht Teil eines Standard Computers. Der zweite und wahrscheinlich größte Nachteil ist die Handhabung des Schlüsselstroms: Jedes Schlüsselstrombit s_i wird nur einmal verwendet. Das One-Time-Pad benötigt somit einen Schlüssel, welcher genau-

so lang ist wie die Nachricht selbst. Es wird klar, warum diese Anforderung selbst bei mittelgroßen Nachrichten problematisch wird, zusätzlich muss nach jeder Übertragung ein neuer Schlüssel ausgetauscht werden. Die Idee von OTP ist gut, jedoch müssen für den praktischen Gebrauch einige Modifikationen vorgenommen werden. Es wird versucht, den Schüsselstrom, welcher aus einer echt zufälligen Quelle stammt, durch die Ausgabe eines CSPRNG zu ersetzen, wobei der Schlüssel k den Startwert des PRNG bildet. Das Prinzip dieser Idee ist in Abbildung 2.4 zu sehen. Zu nächst ist wichtig festzuhalten, dass Stromchiffren nicht beweisbar sicher sind. Es ist allerdings so, dass alle in der Praxis verwendeten Algorithmen (Stromchiffren, Blockchiffren, asymmetrische Verfahren), keine informationstheoretische Sicherheit aufweisen (Paar und Pelzl 2010, S. 38). Um den Sicherheitsbegriff realistischer zu gestallten, wird der Angreifer in seiner Rechenleistung beschränkt, man hofft also, dass ein Verfahren berechenbarkeitstheoretisch sicher ist. Diese Eigenschaft ist von Paar und Pelzl folgendermaßen definiert (ebd., S. 35):

Definition 2.4.3 (*Computational Security*)**.** „*A crpytosystem is computationally secure if the best known algorithm for breaking it requires at least t operations.*“

Literaturverzeichnis

Barry, Mark (Juni 2004). *Cryptography in Home Entertainment*. Einsichtsname: 03.01.2021. URL: http://www.math.ucsd.edu/~crypto/Projects/MarkBarry/index.htm.

Haahr, Mads (2021). *Introduction to Randomness and Random Numbers*. Einsichtsname: 14.01.2021. URL: https://www.random.org/randomness.

Paar, Christof und Jan Pelzl (2010). *Understanding Cryptography*. Springer. ISBN: 978-3-642-04100-6.

Remmert, Reinhold und Peter Ullrich (2008). *Elementare Zahlentheorie*. 3. Aufl. Birkhäuser. ISBN: 978-3-7643-7730-4.