FACHHOCHSCHULE **TRIER**
Hochschule für Technik, Wirtschaft und Gestaltung
University of Applied Sciences
Informatik

Kryptanalyse der doppelten Spaltentranspositionschiffre

Entwurf und Implementierung eines Known-Plaintext Angriffs

Cryptanalysis of the Double Transposition Cipher

Design and Implementation of a Known Plaintext Attack

Tim Wambach, B.Sc.

Master Abschlussarbeit

Betreuer: Prof. Dr. Konstantin Knorr

Trier, 31.05.2011

Zitat

“Perfektion ist nicht dann erreicht, wenn es nichts mehr hinzu zu fügen gibt, sondern wenn man nichts mehr weglassen kann.“
- Antoine de Saint-Exupéry (1900-1944)

Für meine Eltern.

Kurzfassung

Kryptologie, also die Verschlüsselung von Nachrichten, ist in der heutigen Zeit ein elementarer Bestandteil unseres Alltags geworden. Weder im öffentlichen noch im privaten Bereich kann auf die Anwendung kryptologischer Verfahren verzichten werden und sie spielen auch für das Militär eine wichtige Rolle. Während heutzutage fast ausschließlich rechnergestützt gearbeitet wird, musste in den Jahrhunderten vor Entwicklung elektronischer Datenverarbeitung auf einfachere Verfahren zurückgegriffen werden. Eine wichtige Anwendung der Verschlüsselungstechnik im 19. und frühen 20. Jahrhundert war dabei der sogenannte Doppelwürfel (Übchi); ein Verfahren mit einer zweifach angewendeten Spaltentransposition. Dieses Verfahren ist von Hand durchführbar und bis heute ist kein Ciphertext-only Angriff bekannt geworden. In dieser Arbeit wird erstmals eine lineare Kryptanalyse für das Verfahren durchgeführt. Aus diesen Erkenntnissen wird eine Anwendung implementiert, die einen automatisierten Known-Plaintext Angriff auf die Verschlüsselung ermöglicht. So lässt sich bereits mit einem Bruchteil des Klartextes der Schlüssel zurückrechnen. Eigenschaften, die der Chiffre ursprünglich ihre Sicherheit garantieren sollten, werden hier ausgenutzt um das Verfahren zu brechen.

Today, cryptology, the encryption of messages, has become a fundamental part of our common life. Neither in the public nor in the private sector you can renounce cryptographic methods; they also play an important role for the military. Whereas today we work almost exclusively computer-based, in the centuries before the formation of electronic data processing one had to resort to more simple operations. An important application of encryption technology in the 19th and early 20th century is the so-called doppelwuerfel (Übchi), a process where a double columnar transposition is applied. This process is manually actable and up to this day no ciphertext-only attack is known. In this work it is the first time that a linear cryptanalysis will be performed on that process. Based on those findings an application will be implemented, which makes an automated known-plaintext attack possible on the encryption. Thus, already with a fraction of the plaintext the key can be recalculated. Properties, that should guarantee the security of the cipher, are utilized to break the process here.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1

Einleitung

In dieser Arbeit wird der Doppelwürfel bzgl. seiner Sicherheit überprüft. Diese ist davon abhängig, wie leicht sich ein Chiffretext ohne Kenntnis des geheimen Schlüssels dechiffrieren lässt. Gemäß dem Prinzip von Kerckhoff muss die Sicherheit eines Verfahrens lediglich von der Geheimhaltung des Schlüssels abhängen. Das bedeutet, dass Analysen, um welche Art von Verschlüsselung es sich handelt, nicht notwendig sind.

Angriffe auf Verschlüsselungen lassen sich in verschiedene Arten bzw. Stufen unterteilen, welche ein unterschiedliches Maß an Komplexität aufweisen. So ist die erste Stufe der so genannte Chosen Plaintext Angriff. Dabei kann sich der Angreifer beliebige Klartext/Chiffretext Paare generieren lassen, die mit dem selben geheimen Schlüssel chiffriert wurden. Zwar wirkt dies auf den ersten Blick realitätsfern, allerdings kann es durchaus vorkommen, dass z.B. durch eine Schwachstelle in einer Applikation das Interface zur Ver- und Entschlüsselung offengelegt und so die Erstellung von selbst gewählten Paaren möglich ist. Der Angreifer besitzt in diesem Fall maximale Freiheiten.

Die nächste Stufe ist der so genannte Known Plaintext Angriff. Dabei ist der Angreifer im Besitz von Klar-/Chiffretextpaaren, deren Anzahl begrenzt sind und dessen Inhalt nicht selbst gewählt wurden. Ein Beispiel hierfür wäre das Abfangen von Nachrichten, deren Inhalt zu einem späteren Zeitpunkt oder über einen anderen Nachrichtenkanal bekannt wird. Ziel ist es, den verwendeten Schlüssel zu erfahren und ggf. für andere Anwendungen oder Entschlüsselungen nutzbar zu machen.

Am Ende dieser Liste steht der Ciphertext only Angriff, welcher zugleich auch die schwierigste Aufgabe für den Kryptanalytiker darstellt. Hierbei soll es nur aufgrund eines Chiffretextes ermöglicht werden, den Klartext oder den verwendeten Schlüssel zu generieren. Ein Beispiel dafür wäre der Kasiski-Test bei der Entschlüsselung einer Vigenère-Verschlüsselung.

Bevor in diesem Kapitel auf die eigentliche Problemstellung und die Abgrenzung zu anderen Arbeiten eingegangen wird, soll zunächst der Aufbau der Chiffre bzw. ihr Ablauf dargelegt werden. Damit kann sich der Leser zunächst ein Bild über die Verschlüsselung verschaffen. Am Ende dieses Kapitels wird der Aufbau der Arbeit beschrieben, so dass sich der Leser einen Überblick über den Inhalt verschaffen kann.

1.1 Funktionsweise der Chiffre

In diesem Abschnitt soll der Ablauf der Chiffre verdeutlicht werden. So wird ein Beispieltext zunächst ver- und anschließend entschlüsselt.

1.1.1 Verschlüsselung

Die Funktionsweise der Verschlüsselung wird nun an einem Beispiel demonstriert. Zunächst benötigen wir einen Klartext, der verschlüsselt werden soll. Der Klartext lautet an dieser Stelle:„HALLODASHIERISTEINLANGERBEISPIELTEXTUMDASVERFAHREN Ebenso werden zwei Schlüssel benötigt. Als ersten Schlüssel verwenden wir *NOTEBOOK*, dessen Länge 8 beträgt. Nun wird der Klartext zeilenweise in ein Rechteck eingetragen, dessen Breite von der Länge des angewendeten Schlüssels bestimmt wird. Die Höhe des Würfels (also die vertikale Seitenlänge) ist abhängig von der Länge des verwendeten Textes.

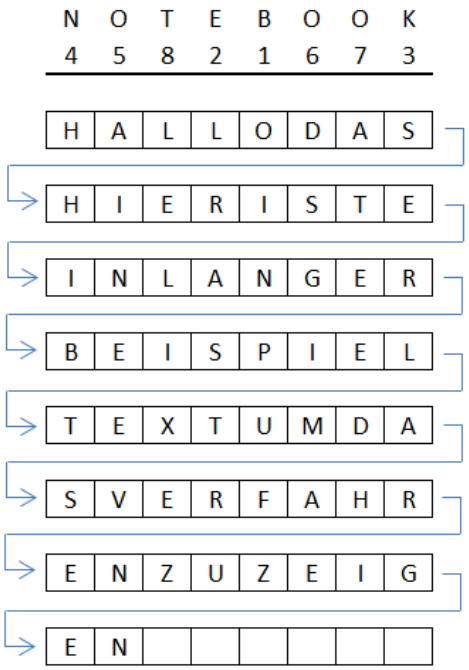


Abbildung 1.1: Zeilenweises Eintragen des Klartextes in den Würfel

Wie zu sehen, ist die letzte Zeile nicht vollständig gefüllt. Dies wäre nur dann der Fall, wenn die Schlüssellänge ein ganzzahliger Teiler der Klartextlänge ist. Anschließend wird der Schlüssel der Chiffre alphabetisch sortiert. Es muss sich dabei um eine stabile Sortierung handeln, was bedeutet, dass die Reihenfolge der Spalten bewahrt wird sofern dessen Schlüsselzeichen identisch sind. Es werden allerdings dabei nicht nur die Schlüsselzeichen vertauscht, sondern auch die darunterliegende Spalte. Dieser Eigenschaft verdankt die Spaltentranspositionschiffre ihren Namen.

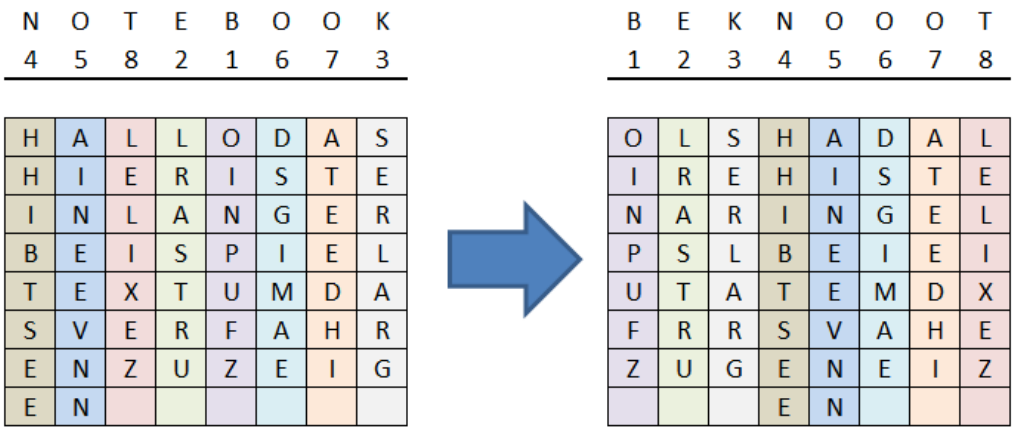


Abbildung 1.2: Spaltentausch in der ersten Runde

Wie zu erkennen, hat sich die Position der Lücken verändert - dies kann allerdings ignoriert werden. Andernfalls wäre das Auffüllen mit Füllzeichen möglich, die nicht im Klartextalphabet enthalten sind, wobei diese am Ende auch wieder entfernt werden können.

Essentiell für den Doppelwürfel ist der nächste Schritt: Das spaltenweise Auslesen der entstandenen Matrix. Das Ergebnis wird also nicht in Lesereihenfolge entnommen, sondern spaltenweise.

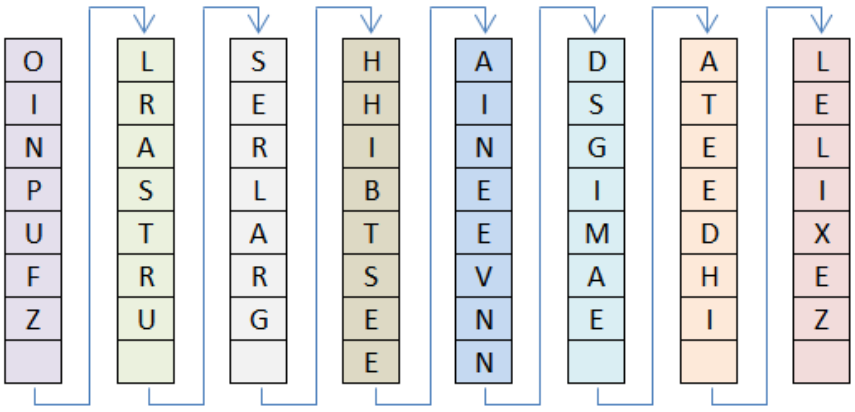


Abbildung 1.3: Spaltenweises Auslesen des Würfels der ersten Runde

Hierbei werden die resultierenden Zeichen (falls noch nicht geschehen) in Großbuchstaben verwandelt und in 5er-Blöcke gruppiert. Dies soll die Lesbarkeit der Chiffrierten Nachricht erhöhen und wurde in [Kri13] empfohlen. Nicht berücksichtigt werden des Weiteren die entstandenen Lücken der letzten Zeile: *OINPU FZLRA STRUS ERLAR GHHIB TSEEA INEEV NNDSG IMAEA TEEDH ILELI XEZ*

Zusammenfassend wurden der Klartext eingetragen, die Spalten in Abhängigkeit eines Schlüssels vertauscht und das Ergebis spaltenweise ausgelesen. Um die

Sicherheit der Chiffre zu verstärken, muss dieser Vorgang - unter Verwendung eines zweiten unabhängigen Schlüssels - erneut durchgeführt werden. In diesem Beispiel verwenden wird den Schlüssel *DECKEL* und es ergibt sich ein neues Rechteck der Breite 6. Der Klartext der zweiten Runde, ist der Chiffretext der ersten und dieser wird zeilenweise in den Würfel eingetragen. Anschließend wird erneut die Spaltentransposition durchgeführt, wie sie bereits für die vorherige Runde beschrieben wurde.

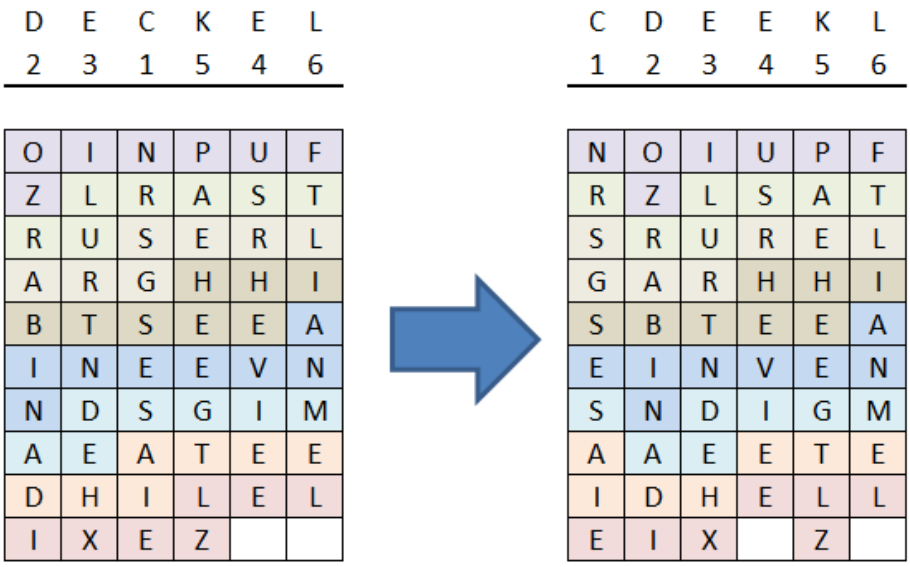


Abbildung 1.4: Zeilenweises Eintragen des Klartextes und spaltenweises Auslesen

Nachdem der Würfel spaltenweise permutiert wurde, kann man das Ergebnis der zweiten Runde auslesen. Wie zuvor wird erneut spaltenweise vorgegangen, um den resultierenden Chiffretext zu gewinnen.

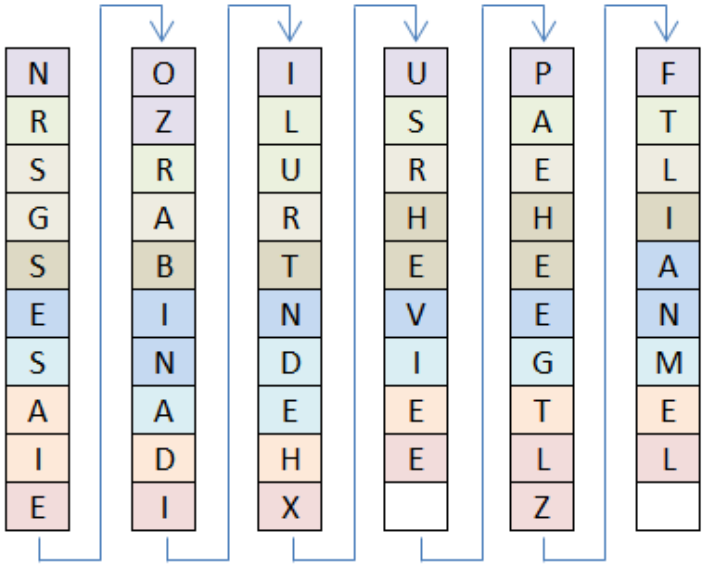


Abbildung 1.5: Spaltenweises Auslesen des Würfels der zweiten Runde

Der Klartext wurde nun erfolgreich in den Chiffretext *NRSGS ESAIE OZRAB INADI ILURT NDEHX USRHE VIEEP AEHEE GTLZF TLIAN MEL* verschlüsselt und kann übermittelt werden.

1.1.2 Entschlüsselung

Nun soll der entstandene Chiffretext wieder entschlüsselt, also in den Klartext verwandelt werden. Dabei werden die Schritte der Verschlüsselung rückwärts ausgeführt. Während der Durchführung der Verschlüsslung wurde gezeigt, dass in der letzten Zeile der Matrix Lücken entstehen bzw. die Matrix nicht vollständig gefüllt ist. Aufgrund der daraus resultierenden Verschiebung gestaltet sich der Vorgang der Entschlüsselung etwas weniger intuitiv, ist jedoch trotzdem verlustfrei durchführbar.

So muss zunächst ein leerer Würfel der zweiten Runde aufgestellt werden, wobei die nicht verwendeten Stellen der letzten Zeile (der Rest) unkenntlich gemacht werden müssen:

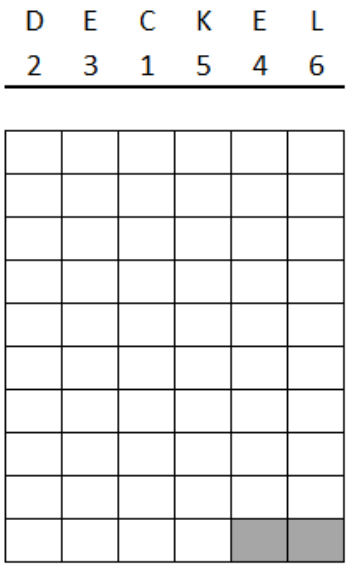


Abbildung 1.6: Entschlüsselung des Chiffretextes in der ersten Runde

Nun kann der Chiffretext spaltenweise in die Matrix eingetragen werden. Die Reihenfolge der Eintragung entspricht der Reihenfolge der Schlüsselzeichen im Alphabet. Das Entschlüsselungsergebnis der ersten Runde ist das zeilenweise Auslesen des entstandenen Würfels.

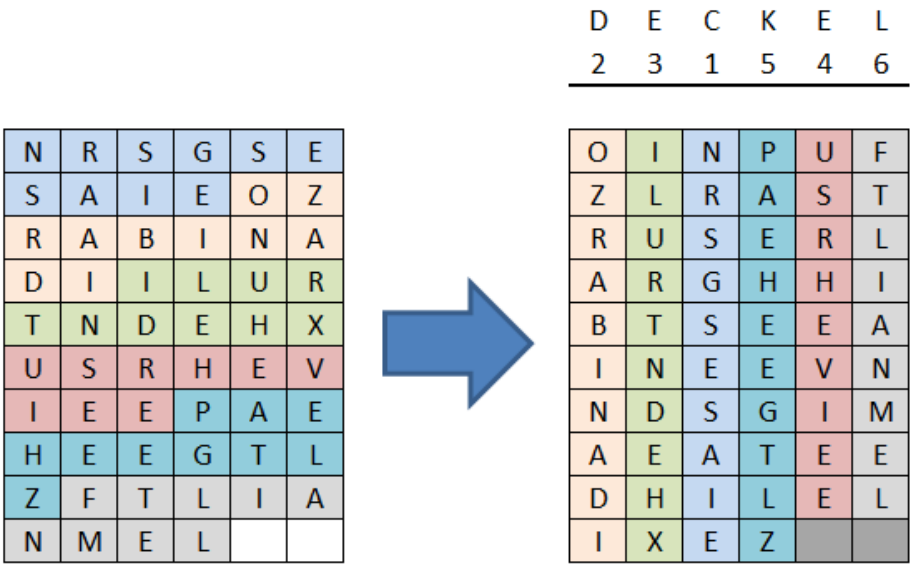


Abbildung 1.7: Spaltenweises Eintragen des Chiffretextes in den Würfel in der ersten Entschlüsselungsrunde

Analog kann dieser Vorgang für die zweite Entschlüsselungsrunde mit dem ersten Schlüssel der Verschlüsselung durchgeführt werden. Wichtig ist hierbei erneut, dass die Spalten mit Überlänge bzw. die Spalten mit fehlenden Feldern in der letzten Zeile beachtet werden.

Als Resultat kann der Klartext nun wieder zeilenweise aus dem Würfel entnommen und gelesen werden.

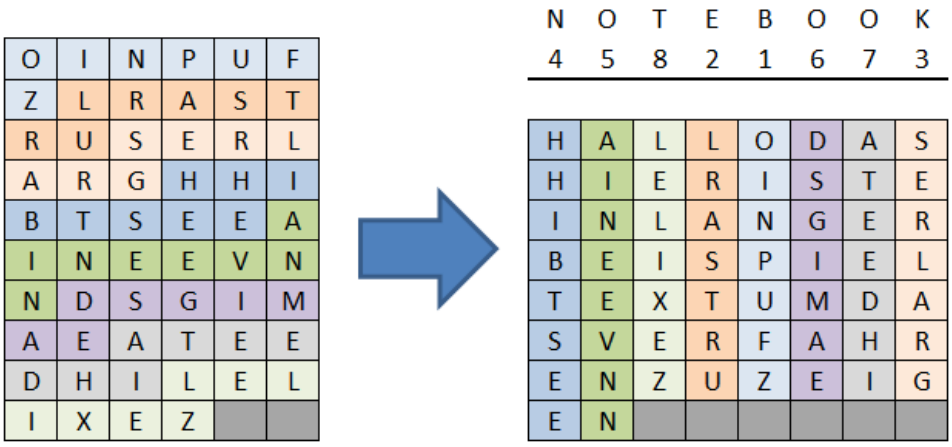


Abbildung 1.8: Zweite Entschlüsselungsrunde und Wiederherstellung des Klartextes

Damit wurde ein Text beispielhaft ver- und wieder entschlüsselt.

1.2 Historische Informationen

Es liegen leider nur wenige Informationen darüber vor, zu welchen Zeitpunkten und welche Art von Informationen mittels Doppelwürfel verschlüsselt wurden. [Sch08] bietet einen umfangreichen Überblick zu Verschlüsselungen aus dieser Zeit, jedoch nur wenig Aufschluss über die Verwendung des Doppelwürfels. Gesichert scheint die Information, dass diese Chiffre bis 1914 eingesetzt wurde, vgl. [Kri13]. Durch [Bec09] ist weiterhin bekannt, das diese Verschlüsselung offensichtlich auch in den Anfängen des zweiten Weltkrieges verwendet wurde. Das Verfahren ist auch unter dem Namen Übchi bekannt. Dieser stammt daher, das chiffrierte Botschaften häufig mit einem „CHI“ und Übungsversuche, die für Kryptoanalytiker von besonderem Interesse waren, entsprechend mit „UEBCHI“ begonnen haben.

Wichtig an dieser Stelle ist die Abgrenzung zum Doppelkasten. Dabei handelt es sich um eine Verschlüsselung, die ein Spezialfall der Playfair-Cipher ist. Dies ist eine bekannte Paar-Substitutionschiffre und hat nichts mit dem Doppelwürfel gemeinsam, wurde allerdings schon erfolgreich analysiert. Häufig werden diese

beiden Methoden jedoch aufgrund der Namensähnlichkeit gleichgesetzt, wodurch es schwierig ist, die überlieferten Informationen voneinander zu trennen. Im nächsten Abschnitt wird auf Arbeiten bzgl. der Kryptanalyse des Doppelwürfels eingegangen.

1.3 Verwandte Arbeiten

Eine sehr wesentliche Arbeit stellt [Kul34] dar, die an dieser Stelle zuerst genannt werden sollte. Dieses 23-seitige Werk enthält Informationen und Beispiele zur Kryptanalyse des Doppelwürfels. Allerdings wird sich hierbei häufig auf Spezialfälle beschränkt, die bei der Anwendung überlicherweise nicht vorkommen. Als Beispiel ist hier die Verwendung identischer Schlüssel für beide Runden zu nennen. Erst im letzten Abschnitt wird auf 5 Seiten ein Chosen-Plaintext Angriff beschrieben, der allerdings nur sehr knapp, generisch und nicht ohne weiteres auf alle Arten von Geheimtexten anwendbar ist. Dieses Buch wurde erst in den 90er Jahren von der NSA[1] deklassifiziert und für die Öffentlichkeit freigegeben.

Chronologisch nicht weit entfernt, findet sich die 4. Ausgabe von [Fri41]. Der bekannte Kryptologe William F. Friedman hat hier Transpositionschiffren im Allgemeinen beschrieben. Das Verfahren des Doppelwürfels wird hier ebenfalls erklärt, jedoch nicht weiter vertieft.

Mit einem Sprung in die 60er Jahre kommen wir zu [Bar95], welches allerdings erst 1995 veröffentlicht wurde. Innerhalb dieses Buches wird ebenfalls auf die zwei vorher genannten Werke verwiesen. Wie schon erwähnt, wurden in den 90ern viele Dokumente freigegeben, was laut Vorwort dieses Buches entscheidend für seine Veröffentlichung war. Das Buch teilt das Problem in die Bereiche auf, welche die Wiederherstellung des Schlüssels

- aus einer gesamten Nachricht,
- mit bekanntem Anfang,
- mit bekanntem Ende,
- mit bekanntem Inhalt an einer bekannten Stelle im Text

ermöglichen soll. Dabei werden im Wesentlichen Bigramme bzw. nebeneinanderliegende Buchstabenpaare betrachtet und wie sich diese innerhalb der Verschlüsselung verschieben. Das Buch ist sehr umfangreich und zeigt gute Ansätze um dem Doppelwürfel entgegenzutreten und die Eigenschaften der Verschlüsselung zu verstehen. Allerdings zeigt sich bei genauer Betrachtung, dass der Autor überwiegend Beispiele und stets den richtigen Lösungsweg genommen hat um das Ziel zu erreichen. Dies auch, obwohl es an manchen Stellen eine große Anzahl von Alternativen gegeben hätte. So wird ebenfalls an keiner Stelle eine formale Definition getroffen, welche den Vorgang der Verschlüsselung oder sogar der Kryptanalyse beschreibt und damit eine Verallgemeinerung zur Entschlüsselung beliebiger Texte ermöglicht.

[1] National Security Agency

Im Jahre 1999 hat der damalige Präsident des BSI[2] im Magazin „Spektrum der Wissenschaft“ einen Artikel (Abbildung 1.10) über kryptologische Verfahren veröffentlicht. Unter anderem wurde hier auch auf den Doppelwürfel eingegangen. In seinem Text rief der Autor auch zu mehr wissenschaftlichen Veröffentlichungen zu diesem Thema auf. Insbesondere erwähnte er das Fehlen eines Ciphertext-only Angriffs auf diese Verschlüsselung.

Um dieser Bitte Nachdruck zu verleihen, veröffentlichte Klaus Schmeh in [Sch08] einen Chiffretext, der mit zwei unabhängigen Schlüsseln der Länge 20-25 verschlüsselt wurde. Sowohl der Text, als auch die Schlüsselwörter, sind in englischer Sprache. Bisher gilt dieses Problem als ungelöst.

Wie bereits beschrieben, beschäftigen sich die meisten Kryptanalysewerke zu Permutationschiffren mit einer einfachen Spaltentranspositionschiffre. Einen guten Überblick bieten [Sin66], [Sta07] und [Sal05]. Innerhalb der sonstigen Literatur finden sich Attacken, die auf das multiple Anagrammieren zurück gehen. Dabei wird ein Known-Plaintext Angriff gefahren, bei dem man zwei Texte gleicher Länge benötigt, die außerdem mit dem selben Schlüssel verschlüsselt wurden. Nun baut man mit diesen Informationen einen Graphen bzgl. beiden Chiffretexten auf und sucht Wörter die im Klartext enthalten sind. Weitere Informationen finden sich in [Bau07] aus der auch Abbildung 1.9 entnommen wurde. In dieser können die Wörter „CLOWN“ und „NIGHT“ zusammengesetzt werden.

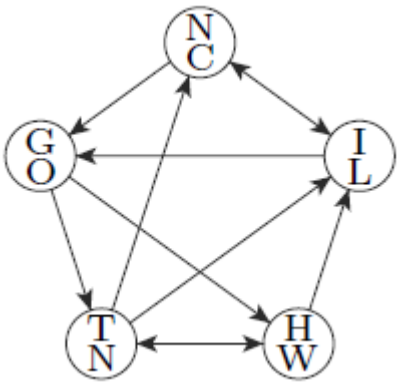
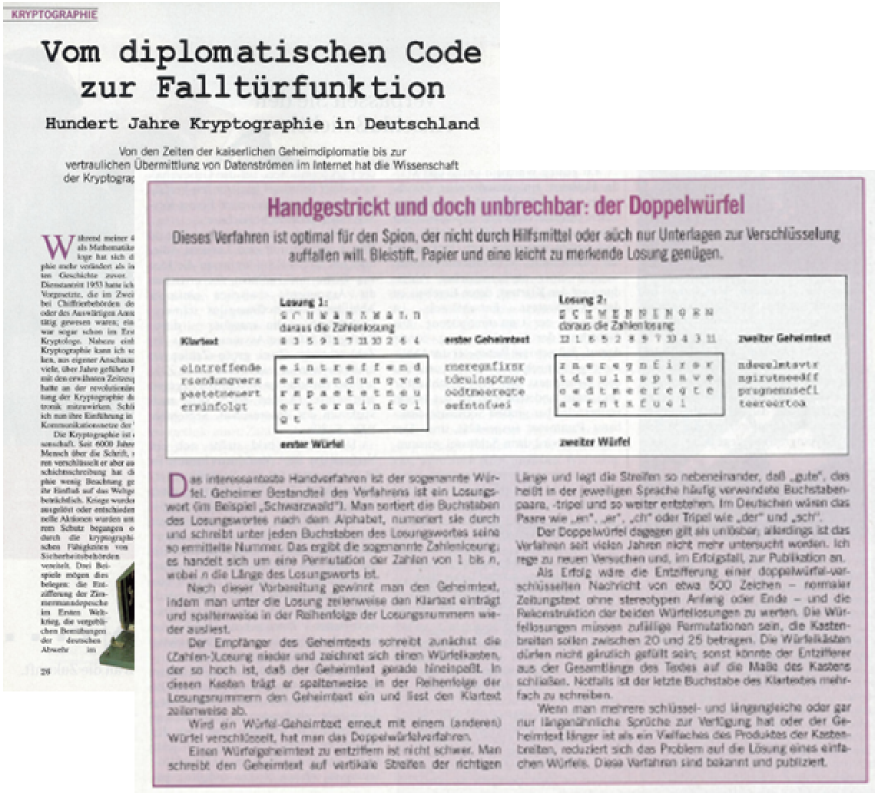


Abbildung 1.9: Multiples Anagrammieren

Diese Angriffsmethode ist sehr effektiv bzgl. der einfachen Spaltentransposition - ist jedoch auch auf den Doppelwürfel anwendbar. Hierbei zeigt sich jedoch eine deutlich höhere Komplexität des entstandenen Graphen. Methoden wie Simulated Annealing[3] optimieren diesen Prozess. Allerdings werden bei dieser Methode mehrere (mindestens zwei) Klar-/Chiffretext-Paare benötigt und sie grenzt sich daher von einem Verfahren ab, bei dem nur ein Klartext bzw. ein Teil davon vorliegt.

[2] Bundesamt für Sicherheit in der Informationstechnik
[3] http://www.springerlink.com/content/n456h888mmj875p8/



KRYPTOGRAPHIE

Vom diplomatischen Code zur Falltürfunktion

Hundert Jahre Kryptographie in Deutschland

Von den Zeiten der kaiserlichen Geheimdiplomatie bis zur vertraulichen Übermittlung von Datenströmen im Internet hat die Wissenschaft der Kryptograp

Handgestrickt und doch unbrechbar: der Doppelwürfel

Dieses Verfahren ist optimal für den Spion, der nicht durch Hilfsmittel oder auch nur Unterlagen zur Verschlüsselung auffallen will. Bleistift, Papier und eine leicht zu merkende Losung genügen.

erster Würfel

zweiter Würfel

Das interessanteste Handverfahren ist der sogenannte Würfel. Geheimer Bestandteil des Verfahrens ist ein Losungswort (im Beispiel „Schwarzwald“). Man sortiert die Buchstaben des Losungswortes nach dem Alphabet, numeriert sie durch und schreibt unter jeden Buchstaben des Losungswortes seine so ermittelte Nummer. Das ergibt die sogenannte Zahlenlosung; es handelt sich um eine Permutation der Zahlen von 1 bis n, wobei n die Länge des Losungsworts ist.

Nach dieser Vorbereitung gewinnt man den Geheimtext, indem man unter die Losung zeilenweise den Klartext einträgt und spaltenweise in der Reihenfolge der Losungsnummern wieder ausliest.

Der Empfänger des Geheimtexts schreibt zunächst die (Zahlen-)Losung nieder und zeichnet sich einen Würfelkasten, der so hoch ist, daß der Geheimtext gerade hineinpaßt. In diesen Kasten trägt er spaltenweise in der Reihenfolge der Losungsnummern den Geheimtext ein und liest den Klartext zeilenweise ab.

Wird ein Würfel-Geheimtext erneut mit einem (anderen) Würfel verschlüsselt, hat man das Doppelwürfelverfahren.

Einen Würfelgeheimtext zu entziffern ist nicht schwer. Man schreibt den Geheimtext auf vertikale Streifen der richtigen Länge und legt die Streifen so nebeneinander, daß „gute“, das heißt in der jeweiligen Sprache häufig verwendete Buchstabenpaare, -tripel und so weiter entstehen. Im Deutschen wären das Paare wie „en“, „er“, „ch“ oder Tripel wie „der“ und „sch“.

Der Doppelwürfel dagegen gilt als unlösbar, allerdings ist das Verfahren seit vielen Jahren nicht mehr untersucht worden. Ich rege zu neuen Versuchen und, im Erfolgsfall, zur Publikation an.

Als Erfolg wäre die Entzifferung einer doppelwürfel-verschlüsselten Nachricht von etwa 500 Zeichen – normaler Zeitungstext ohne stereotypen Anfang oder Ende – und die Rekonstruktion der beiden Würfellosungen zu werten. Die Würfellosungen müssen zufällige Permutationen sein, die Kastenbreiten sollen zwischen 20 und 25 betragen. Die Würfelkästen dürfen nicht gänzlich gefüllt sein; sonst könnte der Entzifferer aus der Gesamtlänge des Textes auf die Maße des Kastens schließen. Notfalls ist der letzte Buchstabe des Klartextes mehrfach zu schreiben.

Wenn man mehrere schlüssel- und längengleiche oder gar nur längenähnliche Sprüche zur Verfügung hat oder der Geheimtext länger ist als ein Vielfaches des Produktes der Kastenbreiten, reduziert sich das Problem auf die Lösung eines einfachen Würfels. Diese Verfahren sind bekannt und publiziert.

Abbildung 1.10: Artikel über den Doppelwürfel von Otto Leiberich

1.4 Problemstellung und Zielsetzung

Der einführende Abschnitt hat die Funktionsweise der Chiffre gezeigt: Durch strukturierte Vertauschung des Klartextes wird der Chiffretext erzeugt. Die farblichen Markierungen in Abbildung 1.5 machen deutlich, das diese Verschlüsselung eine deutlich höhere Diffusion erreicht, als die einfache Spaltentransposition. Daher ist eine Entschlüsselung ohne Kenntnis des Schlüssels als nicht-trivial anzusehen. Wie bei vielen anderen Verschlüsselungen, werden im Doppelwürfel Passwörter als Schlüssel verwendet. An dieser Stelle wäre es möglich, einen Wörterbuchangriff zu fahren. Bei zufälligen Permutationen würde dieser Ansatz jedoch fehlschlagen. Eine vollständige Schlüsselsuche wäre bereits ab einer Schlüssellänge von 15 (für beide Runden) ohne Hoffnung auf Erfolg, da dies einer effektiven Schlüssellänge von 80 Bit entspricht. Diese zwei Möglichkeiten stellen also keinen zufriedenstellenden Lösungsweg dar.

In Abschnitt 1.3 wurde bereits ein Überblick über die bestehenden Arbeiten gegeben. Nach umfangreichen Recherchen konnte keine Software gefunden werden, die einen automatisierten Angriff auf dieses zweirundige Verschlüsselungsverfahren durchführen kann. Diese Einschätzung wird dadurch verstärkt, das innerhalb der Hauptliteratur keine mathematische Betrachtung der Chiffre vorgenommen wurde, die jedoch für die Konzeptionierung eines solchen Algorithmus notwendig ist.

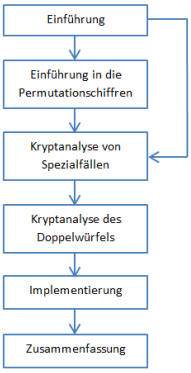
Diese Arbeit soll zunächst einen Überblick über die Verschlüsselung bieten und verschiedene Herangehensweisen demonstrieren. Die dabei entstehenden mathematischen Betrachtungen können anschließend der Krpytanalyse dienen - im Vordergrund steht dabei die Kryptanalyse des allgemeinen Doppelwürfels. So soll effizient ein Chosen Plaintext Angriff aufgebaut werden, der anschließend zu einem Known Plaintext Angriff erweitert wird. Sobald die theoretischen Grundlagen geschaffen worden sind, wird diese Anwendung auch implementiert und bzgl. ihrer Laufzeiteigenschaften getestet.

Teil dieser Arbeit ist also eine Betrachtung der Chiffre, um daraus einen Algorithmus für einen automatisierten Angriff abzuleiten. Da sich zahlreiche Implementierungen finden lassen, die unter anderem auch als Aufgabe für Programmiereinsteiger gilt, ist die Implementierung der Chiffre nicht Teil dieser Arbeit. Da sie aus heutiger Sicht keinen praktischen Nutzen mehr erfüllt, hätte ein Vergleich verschiedener Implementierungsmethoden lediglich akademischen Wert. Aus Gründen der Vollständigkeit wird in dieser Arbeit dennoch ein Lösungsweg für dieses Problem skizziert. Zusammengefasst lauten die Ziele:

- Eine Übersicht der Chiffre bieten und zu anderen Verschlüsselungen abgrenzen.
- Den mathematischen Hintergrund von Permutationschiffren durchleuchten.
- Suche nach Ansätzen zur Kryptanalyse der Verschlüsselung.
- Spezifizierung eines Chosen-Plaintext und Known-Plaintext Angriffs.
- Implementierung der Kryptanalyse in der Programmiersprache Java.
- Bereitstellung einer grafischen Oberfläche als Java-Applet.

1.5 Aufbau der Arbeit

Nachdem in diesem Kapitel das Thema eingeleitet wurde, wird in Kapitel 2 die Funktionsweise der Chiffre vertieft. Sie bietet erste Ansätze, der Verschlüsselung mathematisch entgegenzutreten.



Auch wenn der Inhalt nicht direkt innerhalb der Kryptanalyse verwendet wird, sind die dort gewonnen Kenntnisse für das Gesamtverständnis der Verschlüsselung wichtig. Falls jedoch ein sofortiger Einsteig in die Kryptanalyse gewünscht wird, kann Kapitel 2 übersprungen werden.

Für den Leser möglicherweise befremdlich, werden in dieser Arbeit zunächst Spezialfälle betrachtet, bevor sich dem allgemeinen Doppelwürfel genähert wird. Als ein solcher Spezialfall ist der quadratische und der vollständig gefüllte Doppelwürfel anzusehen. Diese Fälle stellen für die Kryptanalyse ein geringeres Problem dar und werden daher in Kapitel 3 behandelt. In Kapitel 4 wird schließlich der allgemeine Doppelwürfel betrachtet und analysiert. Dabei werden im Wesentlichen die aus dem vorherigen Kapitel gewonnenen Informationen für diesen allgemeinen Fall erweitert. Schrittweise wird sich über ein Chosen-Plaintext-Angriff dem Known-Plaintext-Angriff angenähert.

Der Aufbau der Arbeit entspricht deshalb einem Vorgehen vom Einfachen zum Schwierigen.

Aus den Erkenntnissen von Kapitel 3 und 4 wird ein Algorithmus konstruiert, dessen Implementierung in Kapitel 5 dokumentiert wird. Die Ergebnisse der Arbeit werden abschließend in Kapitel 6 zusammengefasst.

2

Einführung in die Permutationschiffren

Wie innerhalb der Einleitung schon erläutert, handelt es sich beim Chiffretext der doppelten Spaltentranspositionschiffre um eine Permutation des Klartextes. Durch Vertauschung von Plätzen wird der Chiffretext aus dem Klartext zusammengesetzt. Diese ist allerdings nicht willkürlich, sondern folgt einem konkreten Muster, welches über den Schlüssel definiert wird. Hierbei taucht der Begriff Permutation auf. Ein Thema, das im ersten Abschnitt dieses Kapitels genauer betrachtet wird. Anschließend werden Permutationen in Chiffrierverfahren allgemein betrachtet, um sich dann dem eher simplen Einfachwürfel - also der einfachen spaltentranspositionschiffre zu nähern. Diese bildet eine Runde des Doppelwürfelverfahrens das abschließend behandelt wird.

Dieses Kapitel dient dem guten Verständnis der einfachen bzw. doppelten Spaltentransposition. Die gewonnenen Kenntnisse, insbesondere aus Abschnitt 2.3, werden nicht unmittelbar in den nächsten Kapiteln benötigt, helfen jedoch die Problempunkte der Verschlüsselung zu erfassen und dienen ggf. als Ansatzpunkte für alternative Lösungswege.

2.1 Definition von Permutationen

Innerhalb [JM07] wird der Begriff der Permutation folgedermaßen definiert:

Definition 2.1. *(Permutation) Eine bijektive Abbildung einer endlichen Menge X in sich selbst heißt Permutation der Menge X. Sind die Elemente aus X irgendwie angeordnet, dann können wir eine Permutation als Umsortieren der Elemente von X in einer neuen Reihenfolge auffassen.*

Nehmen wir als Beispiel eine Menge $X = \{x_1x_2x_3x_4\}$, so kann eine Permutation über eine bijektive Funktion $f\colon X \to X$ definiert werden:

$$\begin{aligned} f(x_1) &= x_2 \\ f(x_2) &= x_4 \\ f(x_3) &= x_3 \\ f(x_4) &= x_1 \end{aligned} \tag{2.1}$$

und es entsteht aus X über die Funktion f ein neues $X' = \{x_2x_4x_3x_1\}$. Eine übliche Schreibweise zur Definition einer Permutation stellt die Matrixdarstellung

(nicht zu verwechseln mit einer Permutationsmatrix) dar:

$$f : \begin{pmatrix} x_1\ x_2\ x_3\ x_4 \\ x_2\ x_4\ x_3\ x_1 \end{pmatrix} \tag{2.2}$$

Diese Matrixschreibweise stellt die Zuordnung von Bild- und Zielmenge gegenüber: In der oberen Zeile ist die Bild- bzw. Definitionsmenge abgebildet und in der unteren die entsprechende Zielmenge. Innerhalb der ersten Zeile sind die Elemente in ihrer natürlichen Reihenfolge aufgelistet. Aus diesem Grund wird sich häufig nur auf die zweite Zeile beschränkt:

$$f : \begin{pmatrix} x_2\ x_4\ x_3\ x_1 \end{pmatrix} \tag{2.3}$$

Die Permutationsfunktion war in diesem Fall von $f\colon X \to X$ definiert. Da der Wert, also der Inhalt der Menge, bei der Permutation von keiner Bedeutung ist, können wir diesen vernachlässigen. So definieren wir f als eine Funktion $f\colon \{1,2,3,4\} \to \{1,2,3,4\}$ um und verwenden nur den Index bzw. die absolute Position:

$$\begin{aligned} f(1) &= 2 \\ f(2) &= 4 \\ f(3) &= 3 \\ f(4) &= 1 \end{aligned} \tag{2.4}$$

Daraus folgt die Anordnung

$$\begin{pmatrix} 2\ 4\ 3\ 1 \end{pmatrix}$$

und wird als **einzeilige Schreibweise** bezeichnet. Neben dieser, gibt es noch weitere Möglichkeiten eine Permutation auszudrücken:

- Zyklenschreibweise,
- die Tupelschreibweise und die
- Permutationsmatrix.

Häufig wird sich auf die einzeilige Schreibweise beschränkt, diese kann jedoch verlustfrei in andere Formen umgewandelt werden. Die neu erworbenen Kenntnisse bzgl. der Permutation, können nun auf das Problem der Chiffre übertragen werden. Unser Klartext $P = p_1 p_2 p_3 p_4 \dots p_s$ kann über eine Permutationsfunktion f in den Chiffretext übertragen werden:

$$C = p_{f(1)} p_{f(2)} p_{f(3)} p_{f(4)} \dots p_{f(s)} \tag{2.5}$$

Beispielhaft können wir f wie folgt definieren:

$$f(x) =_{def} \begin{cases} x+1 & falls\, x < s \\ 1 & falls\, x = s \end{cases} \tag{2.6}$$

Dies entspricht der folgenden Definition in Matrixschreibweise:

$$\begin{pmatrix} 1\ 2\ 3\ 4\ 5 \dots\ s \\ 2\ 3\ 4\ 5\ 6\ \dots\ 1 \end{pmatrix} \tag{2.7}$$

So wird aus dem Klartext „HALLOWELT“ der Chiffretext „ALLOWELTH“. Damit ist die direkte Übertragung des Klartextes in den Chiffretext möglich.

An dieser Stelle ist es wichtig einzusehen, dass innerhalb von Permutationschiffren häufig eine Permutation als Schlüssel verwendet wird. Der Schlüssel stellt jedoch nicht die oben beschriebene Permutationsfunktion dar, sondern wird, abhängig von der eingesetzten Chiffre, nur zur Definition dieser verwendet.

2.1.1 Verknüpfung von Permutationen

Zwei Permutationen über der gleichen Menge lassen sich durch Verknüpfung (oder Komposition) zu einer einzigen verbinden. Seien Q und R Permutationen in einzeiliger Schreibweise der gleichen Länge t, so existieren die bijektiven Funktionen q und r. Das Ergebnis der Verknüpfung ist die Permutation S mit der bij. Funktion s die definiert ist als:

$$\forall i \in [1,t] : s(i) =_{def} \; q(r(i)) \tag{2.8}$$

Dies soll anhand eines Beispiels visualisiert werden. Hierfür definieren wir zwei Permutationen in zweizeiliger Form:

$$Q = \begin{pmatrix} 1\,2\,3 \\ 3\,1\,2 \end{pmatrix} \; R = \begin{pmatrix} 1\,2\,3 \\ 1\,3\,2 \end{pmatrix} \tag{2.9}$$

Wir wollen nun Q und R zu einer einzigen Matrix verbinden:

$$Q \circ R = S = \begin{pmatrix} 1\,2\,3 \\ ?\,?\,? \end{pmatrix} \tag{2.10}$$

Für die erste Position in S betrachten wir zunächst die hintere Matrix R. Hier suchen wir die 1 innerhalb der unteren Zeile und betrachten, an welcher Position sich diese befindet. Bei unserem Beispiel befindet sich die 1 an der ersten Position in R. Im nächsten Schritt betrachten wir die erste Position in Q, wobei sich dort die 3 befindet. Daraus folgt, dass die 1. Position in S mit 3 belegt wird. Die zweite Position in S finden wir, wenn wir die 2 in R suchen. Diese befindet sich hier an der dritten Position. An der dritten Position in Q finden wir die 2 und so folgt, dass die zweite Position in S mit 2 gefüllt wird. Im letzten Schritt suchen wir die 3 in R und finden diese an der zweiten Position. An der zweiten Position in Q befindet sich die 1 und diese wird folglich in S an die dritte Position gesetzt. Es entsteht:

$$Q \circ R = S = \begin{pmatrix} 1\,2\,3 \\ 3\,2\,1 \end{pmatrix} \tag{2.11}$$

und in einzeiliger Form:

$$\big(3\,1\,2\big) \circ \big(1\,3\,2\big) = \big(3\,2\,1\big) \tag{2.12}$$

2.1.2 Berechnung der Inversen

Für jede Permutation existiert eine zweite, mit deren Verknüpfung das neutrale Element erzeugt wird: die Inverse. Gegeben sei eine Permutation f und gesucht wird die Inverse:

$$f \circ f^{-1} = e \tag{2.13}$$

Zunächst ist es notwendig e zu kennen. Das neutrale Element ist stets eine Permutation in natürlicher Reihenfolge:

$$\big(1\ 2\ 3 \ldots n\big) \tag{2.14}$$

Sei eine Permutation f gegeben, so suchen wir eine Inverse, mit der folgende Gleichung erfüllt wird:

$$\big(f(1)\ f(2)\ f(3) \ldots f(n)\big) \circ \big(f^{-1}(1)\ f^{-1}(2)\ f^{-1}(3) \ldots f^{-1}(n)\big) = \big(1\ 2\ 3 \ldots n\big) \tag{2.15}$$

Durch Unterabschnitt 2.1.1 wissen wir, dass die Verknüpfung der Permutationen Q und R der Länge t die Permutation S entstehen lässt und die Funktion s über die Funktionen q und r definiert wird:

$$\forall i \in [1,t] : s(i) =_{def}\ q(r(i)) \tag{2.16}$$

Sofern die resultierende Permutation S das neutrale Element ist, gilt an dieser Stelle

$$\forall i \in [1,t] : i =_{def}\ q(r(i)) \tag{2.17}$$

Gesucht ist hier also die Permutation R der Länge t für die gilt:

$$r(i) =_{def} j \tag{2.18}$$

wobei $j \in [1,t] : q(j) = i$. Als Beispiel suchen wir die Inverse zur Permutation $Q = \big(3\ 1\ 2\ 4\big)$, so dass gilt:

$$\big(3\ 1\ 2\ 4\big) \circ R = \big(1\ 2\ 3\ 4\big) \tag{2.19}$$

Gemäß der oben gezeigten Definition suchen wir nun nacheinander die Positionen in Q, die der natürlichen Reihenfolge entsprechen und finden:

$$Q^{-1} = R = \big(2\ 3\ 1\ 4\big) \tag{2.20}$$

Eine möglicherweise intuitivere Form der Betrachtung ist die Umsortierung innerhalb der Tupelschreibweise. So wird zunächst die Permutation Q in zweizeiliger Form notiert. Anschließend werden die Spalten in dieser Matrix nach der zweiten Zeile umsortiert. Die Inverse taucht an dieser Stelle nun in der oberen Spalte der resultierenden Matrix auf:

$$\begin{pmatrix} 1\ 2\ 3\ 4 \\ 3\ 1\ 2\ 4 \end{pmatrix} \rightarrow \begin{pmatrix} 2\ 3\ 1\ 4 \\ 1\ 2\ 3\ 4 \end{pmatrix} \tag{2.21}$$

2.1.3 Anzahl von Permutationen

Wie wir gesehen haben, gibt es neben der natürlichen Reihenfolge 1 2 3 4 5 ... noch verschiendene andere Permutationen:

$$\begin{array}{l} 1\,2\,3\,4\,5\ldots \\ 1\,3\,2\,4\,5\ldots \\ 1\,3\,4\,2\,5\ldots \\ 1\,3\,4\,5\,2\ldots \\ 5\,4\,3\,2\,1\ldots \end{array} \tag{2.22}$$

Die Anzahl der möglichen Permutationen in einer n-Elementigen Menge beträgt $n!$. Der Beweis hierfür kann aus [Tes06] entnommen werden.

2.2 Permutationen in Chiffrierverfahren

Es ist offensichtlich, dass es sich bei einem sinnvollen Text - also dem Klartext - um eine geordnete Menge handelt. Den Klartext definieren wir als

$$P = p_1 p_2 p_3 \ldots p_s \tag{2.23}$$

und mittels

$$C = c_1 c_2 c_3 \ldots c_s \tag{2.24}$$

den Chiffretext. Da innerhalb reiner Permutationschiffren lediglich die Position der Zeichen verändert wird, ist die Länge des Klartextes mit der des Chiffretextes identisch und daher gilt:

$$|P| = |C| = s \tag{2.25}$$

Ziel einer Permutationschiffre ist es, die Elemente des Klartextes zu permutieren, so dass ein nicht lesbarer Chiffretext entsteht.

Definition 2.2. *(Permutationschiffre) Innerhalb einer Permutationschiffre wird der Klartext P über eine Permutationsfunktion f (oder π) zum Chiffretext C umsortiert. Es gilt:*

$$|P| = |C|$$

2.2.1 Skytale

Bei der Skytale handelt es sich um eine der ältesten bekanntesten Verschlüsselungsmethoden und eine einfache Form der Transpositionschiffre. Sie wurde von den Spartaner vor über 2000 Jahren eingesetzt, um Botschaften nicht im Klar- sondern in einem Geheimtext zu übermitteln. Der Schlüssel wird hierbei durch den Durchmesser eines Holzstabs geliefert.

Die Idee hinter dieser Verschlüsselung ist das Umwickeln eines Stabs mit Pergament, welches so eine einheitliche, zu beschriftende Oberfläche bietet. Anschließend wird der Text spaltenweise (vertikal) auf den Stab notiert. Nach Abwickeln des Pergamentbandes erhalten wir eine Permutation des Klartextes, welche nur mit einem Stab gleichen Durchmessers wieder in einen lesbaren Text verwandelt werden kann. Eine gelungene Darstellung der Verschlüsselung findet sich in [Beu09] und [Sin09].

2.2.2 Der Gartenzaun

Bei der Gartenzaun-Verschlüsselung nach [Bau07] wird der Klartext P Buchstabenweise abwechselnd in zwei Zeilen geschrieben. So findet sich in der ersten Zeile jeder erste, dritte, fünfte, etc. und in der zweiten Zeile jeder zweite, vierte, sechste etc. Buchstabe. Anschließend wird die zweite Zeile an die erste konkateniert. Tatsächlich handelt es sich bei dieser Verschlüsselung um eine einfache Spaltentranspositionschiffre mit dem Permutationsschlüssel $\begin{pmatrix} 1 & 2 \end{pmatrix}$ wie sie in Abschnitt 2.3 noch beschrieben wird.

2.2.3 ADFGX

Bei diesem Verfahren handelt es sich um eine Verschlüsselung, die vom deutschen Militär im ersten Weltkrieg eingesetzt wurde. Hierbei wird sowohl eine Substitution, als auch eine Transposition angewendet. Das zweistufige Verfahren ersetzt innerhalb der ersten Runde Buchstabenpaare gemäß einem Schlüsselquadrat, ähnlich der Playfair Chiffre. Im zweiten Schritt wird eine einfache Spaltentransposition durchgeführt, ähnlich wie sie in Abschnitt 1.1 beschrieben wurde.

Das Verfahren wird in [Sin01] genauer dargestellt. Den Franzosen ist 1918 die Kryptanalyse dieser Chiffre gelungen.

2.2.4 Fleißnersche Schablone

Bei dieser Verschlüsselung handelt es sich um ein Verfahren, in welches über eine Schablone der Klartext einer Nachricht in einen Chiffretext permutiert wird. Zunächst wird eine Schablone angefertigt. Dabei wird ein quadratisches Stück Stück Papier in kleinere Quadrate eingeteilt, die an zufälligen Stellen ausgeschnitten werden. Legt man nun diese Schablone auf ein Blatt Papier, ergibt sich an den ausgeschnittenen Stellen die Möglichkeit Text bzw. einen Buchstaben einzutragen. Anschließend wird das Quadrat um 90° gedreht und der Vorgang wiederholt. Lücken zwischen den Buchstaben werden mit zufälligen Zeichen aufgefüllt. Die Entschlüsselung erfolgt durch die wiederholte Anwendung der Schablone. Das Verfahren wird in [Kip06] genauer beschrieben.

2.2.5 Transposition in modernen Verfahren

Da sich Permutationschiffren stets linear verhalten, werden in modernen Verfahren auch nicht-lineare Komponenten verwendet. Der Encryption Standard (DES) ist eine so genannte Feistelchiffre welche über mehrere Runden Blöcke in zwei Hälften aufgespaltet und vertauscht (permutiert) wird. Jedoch wird hier durch Anwendung der Feistelfunktion ebenfalls eine nicht-lineare Subsitution vorgenommen. Dies gilt ebenfalls für den Advanced Encryption Standard (AES) welcher über ein Galois-Feld die Substitution von Byte-Blöcken vornimmt.

2.3 Die einfache Spaltentranspositionschiffre

Da der Doppelwürfel lediglich aus einer doppelten Ausführung dieses Verfahrens besteht, stellt die einfache Spaltentransposition daher einen guten Einstiegspunkt in die Verschlüsselung dar. Dieser Abschnitt beschreibt diese einfache Form der Verschlüsselung möglichst genau, um ein Verständnis für den Ablauf und die Schwierigkeiten zu entwickeln. Die gewonnen Informationen werden in dieser Arbeit zwar nicht zur Kryptanalyse verwendet, dienen jedoch der in Kapitel 5 beschriebenen Implementierung.

Definition 2.3. *(Spaltentranspositionschiffre) Die* Spaltentranspositionschiffre *ist eine Permutationschiffre nach Definition 2.2, in welcher der Klartext P in rechteckiger Form (*Würfel*) angeordnet und spaltenweise anhand einer Permutation vertauscht wird. Die Höhe des Würfels ergibt sich aus der Länge des Klartextes. Die Breite des Würfels wird über die Länge des Schlüssels K bestimmt.*

Innerhalb der einfachen Spaltentranspositionschiffre wird der Klartext zunächst in Form eines Rechtecks aufgeschrieben. Die Breite des Rechtecks wird vom Schlüssel festgelegt, welcher in Unterabschnitt 2.3.1 näher beschrieben wird. Die Tiefe des Rechtecks bzw. Würfels ist abhängig von der Länge des Klartextes und unterliegt keiner Beschränkung. Anschließend werden die Spalten des Rechtecks in Abhängigkeit des Schlüssels vertauscht. Das Ergebnis stellt erneut einen Würfel dar, dessen Inhalt nun jedoch nicht zeilen- sondern spaltenweise ausgelesen wird.

Definition 2.4. *(Einfachwürfel) Der* Einfachwürfel *ist eine Spaltentranspositionschiffre, in welcher der Klartext zeilenweise in den Würfel (nach Definition 2.3) eingetragen, spaltenweise permutiert und spaltenweise ausgelesen wird.*

Das Cryptool[1] in der Version 1.4.30, bietete die Möglichkeit eines Known-Plaintext-Angriffs auf diese Verschlüsselung, sofern ein vollständiges Klar/Chiffretextpaar vorliegt. Ein Ciphertext-only Angriff wird durch Bigramm-Analysen vorgenommen. Die Idee ist, bekannte Buchstabenpaare der jeweiligen Sprache (z.B. ie, en, etc.) im Chiffretext zu suchen und die Annahme zu treffen, dass diese im Klartext nebeneinander gelegen haben. Daraus ergeben sich Schlüsselinformationen. Auch das multiple Anagrammieren aus Abschnitt 1.3 ermöglicht eine Kryptanalyse.

2.3.1 Der Schlüssel

Bevor wir das Verfahren genauer betrachten, werden wir uns zunächst dem Schlüssel widmen. Jede nicht triviale Verschlüsselung basiert auf einem Schlüssel. Mit Hilfe dieses Schlüssels lässt sich bei einer symmetrischen Verschlüsselung der Klartext in Chiffretext umwandeln und umgekehrt. In der Praxis ist die Länge des Schlüssels üblicherweise kürzer als die des Textes. So werden bei modernen Chiffren Schlüssellängen von 128-256 Bit eingesetzt, um Terrabyte von Daten zu verschlüsseln. Allein diese Tatsache bietet eine Angriffsmöglichkeit. Beim

[1] http://www.cryptool.org/

Einfachwürfel wird ein Schlüssel benötigt, mit welcher die Spalten des Klartextes permutiert werden. Wie der Name bereits sagt, werden die Spalten hierbei anhand einer eindeutigen Permutation vertauscht.

Um die Verschlüsselung komfortabel zu gestalten, kann aus beliebigen Wörtern eine solche Permutation gebildet werden. Zunächst wird ein Wort oder eine Phrase benötigt, die aus Großbuchstaben ohne Sonder- und Leerzeichen besteht z.B. das Wort *CIPHER*. Danach wird jeder Buchstabe einzeln betrachtet. Sie werden, von links ausgehend, abhängig von ihrer Position im Alphabet durchnummeriert. Da C hier die höchste Position 3 im Alphabet hat, wird dieses mit einer 1 versehen, gefolgt vom E, H, etc. und es entsteht:

$$\begin{pmatrix} C\ I\ P\ H\ E\ R \\ 1\ 4\ 5\ 3\ 2\ 6 \end{pmatrix} \tag{2.26}$$

Durch Nummerierung von links nach rechts wird der Fall berücksichtigt, dass ein Buchstabe doppelt enthalten ist. Wäre hier das C doppelt vorhanden, würde das erste Aufkommen von links die 1 und das zweite die 2 erhalten - das E hätte demnach erst die dritte Position. Damit ist die Generierung einer numerischen Permutation aus einem Textschlüssel abgeschlossen. So entsteht die folgende Permutation in Matrixschreibweise:

$$\begin{pmatrix} 1\,2\,3\,4\,5\,6 \\ 1\,4\,5\,3\,2\,6 \end{pmatrix} \tag{2.27}$$

und äquivalent in einzeiliger Schreibweise $\big(1\,4\,5\,3\,2\,6\big)$.

Durch einen zweiten Text würde sich auf die gleiche Art ein weiterer unabhängiger Schlüssel generieren lassen. Mit Hilfe dieses Verfahrens wurde aus einem beliebigen Text ein so genannter Permutationsschlüssel gewonnen.

Definition 2.5. *(Permutationsschlüssel) Der Schlüssel einer Spaltentranspositionschiffre (Def. 2.3) ist eine Permutation (Def. 2.1) und definiert als:*

$$K = \big(k_1\ k_2\ k_3 \ldots k_{|K|}\big)$$

2.3.2 Funktionsweise

Wie bereits beschrieben, wird der Klartext spaltenweise vertauscht. Das bedeutet, bei Anwendung der Verschlüsselung werden Gruppen des Klartextes in Abhängigkeit des Schlüssels verschoben. Ebenfalls erwähnt wurde, dass der Inhalt des Chiffretextes - also die eigentlichen Zeichen - dabei nicht von Interesse sind. Aus diesem Grund beschränken wir uns erneut auf den Index und so wird der Klartext $P = p_1p_2p_3p_4 \ldots p_s$ als eine Folge von natürlichen Zahlen beschrieben:

$$\big(1\,2\,3\,4 \ldots s\big) \tag{2.28}$$

Dieser soll nun über den Schlüssel K permutiert werden. Gemäß des Verfahrens werden die Klartextzeichen in einer Matrix waagerecht eingetragen - also zeilenweise. Die Breite der Matrix entspricht der Länge des Schlüssels, wogegen die Höhe durch die Länge des Textes begrenzt wird. In der ersten Zeile wird innerhalb der Klammern der verwendete Schlüssel K angegeben. Dabei handelt es sich um die Permutation die direkt oder über einen Textschlüssel generiert wurde.

$$\begin{pmatrix} [k_1] & [k_2] & [k_3] & [k_4] & \dots & [k_n] \\ 1 & 2 & 3 & 4 & \dots & n \\ n+1 & n+2 & \dots & & \dots & 2n \\ \vdots & & & & & \vdots \\ (m-1)*n+1 & & \dots & & & m*n \end{pmatrix} \tag{2.29}$$

Bei zeilenweiser Betrachtung der Matrix erhalten wir zunächst $(m-1)$-viele Gruppen der Länge $n = |K|$, z.B. $\big(1\,2\,3\,4\,\dots\,n\big)$, $\big(n+1\;n+2\;n+3\;n+4\,\dots\,2n\big)$, etc. Die letzte ($m$-te) Zeile dagegen kann, je nach Länge des Klartextes, einen Rest enthalten, muss also nicht vollständig gefüllt sein.

Jede dieser Gruppen muss nun mit Hilfe des Schlüssels K permutiert werden. Das bedeutet, der Klartext wird in Abhängigkeit des Schlüssels umsortiert. Diese Vorgehensweise ist noch aus Unterabschnitt 2.1.2 bekannt, in der die Inverse einer Permutation berechnet wurde. Betrachten wir nun die erste Gruppe. Es handelt sich dabei um die ersten n Zeichen des Klartextes, welche mit Hilfe des Schlüssels permutiert werden. Wenden wir beispielsweise die Permutation

$$\big(3\,1\,2\,4\big) \tag{2.30}$$

auf die erste Textzeile an, so erhalten wir:

$$\big(2\,3\,1\,4\big) \tag{2.31}$$

Wir definieren an dieser Stelle das Resultat als **resultierender Schlüssel**. Dieser Name wurde gewählt, da es ein Abbild des Schlüssels auf den Chiffretext darstellt. Es handelt es sich hierbei um die Inverse des Schlüssels, deren Verknüpfung mit diesem das neutrale Element ergeben würde wie in Unterabschnitt 2.1.2 gezeigt wurde.

Die entstehende Permutation der ersten Zeile im Chiffretext ist also die Umsortierung des Klartextes gemäß dem Schlüssel. Zur Verdeutlichung an dieser Stelle ein einfaches Beispiel. Der Klartext ist 14 Zeichen lang und wird in eine Matrix eingetragen, die über eine Schlüssellänge von 4 definiert wird.

$$\begin{pmatrix} [3] & [1] & [2] & [4] \\ 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & & \end{pmatrix} \tag{2.32}$$

Es ist ersichtlich, dass hier der Schlüssel $K = \begin{pmatrix} 3\,1\,2\,4 \end{pmatrix}$ verwendet wird. Wie beschrieben entnehmen wir die erste Zeile $\begin{pmatrix} 1\,2\,3\,4 \end{pmatrix}$ und wenden auf diese den Schlüssel K an bzw. berechnen K^{-1}.

Das Ergebnis lautet $K^{-1} = \begin{pmatrix} 2\,3\,1\,4 \end{pmatrix}$. Dieses Vorgehen ist zum Verständnis der Chiffre und der späteren Implementierung von Bedeutung. Es ist die Abbildung des Schlüssels auf die Zeilen des Chiffretextes. Wurde die Abbildung an dieser Stelle zunächst nur für die erste Zeile definiert, so kann sie nun auch auf den Rest der Matrix ausgeweitet werden. Zunächst kann dies an einem Beispiel deutlich gemacht werden. Wie schon beschrieben, bildet sich in der ersten Zeile der Matrix K^{-1} ab. Wir erkennen jedoch ebenso, dass die Spalte durch Addition der Schlüssellänge (hier $K = 4$) fortgesetzt wird. Daher kann das Resultat, ungeachtet des Inhaltes, durch folgende Matrix beschrieben werden:

$$
\begin{pmatrix}
[1] & [2] & [3] & [4] \\
2 & 3 & 1 & 4 \\
6 & 7 & 5 & 8 \\
10 & 11 & 9 & 12 \\
14 & & 13 &
\end{pmatrix}
\downarrow
\begin{pmatrix}
[1] & [2] & [3] & [4] \\
2 & 3 & 1 & 4 \\
2+4 & 3+4 & 1+4 & 4+4 \\
2+2*4 & 3+2*4 & 1+2*4 & 4+2*4 \\
2+3*4 & & 1+3*4 &
\end{pmatrix}
\downarrow
\begin{pmatrix}
[1] & [2] & [3] & [4] \\
k_1^{-1} & k_2^{-1} & k_3^{-1} & k_4^{-1} \\
k_1^{-1}+|K| & k_2^{-1}+|K| & k_3^{-1}+|K| & k_4^{-1}+|K| \\
k_1^{-1}+2\,|K| & k_2^{-1}+2\,|K| & k_3^{-1}+2\,|K| & k_4^{-1}+2\,|K| \\
 & k_2^{-1}+3\,|K| & k_3^{-1}+3\,|K| &
\end{pmatrix}
\tag{2.33}
$$

Die Matrix muss nun spaltenweise ausgelesen werden. Eine Gegenüberstellung zwischen Klar- und Chiffretext ist in Tabelle 2.1 zu sehen. Die Zeile P enthält die Positionen des Klartextes und C die des Chiffretextes. Es ist erkennbar, dass die ersten drei Chiffrezeichen durch die erste Stelle des resultierenden Schlüssels K^{-1} also k_1^{-1}, weitere 4 durch die zweite Stelle k_2^{-1}, die folgenden vier durch die dritte Stelle k_3^{-1} und die letzten drei durch die letzte Stelle k_4^{-1} beeinflusst werden. Ebenso ist zu sehen, dass die Länge des Schlüssels unmittelbar in Teile des Ergebnisses einfließt.

Es ist zu beachten, dass es sich bisher nur um die einfache Spaltentransposition gehandelt hat. Bei der doppelten Spaltentranspositionschiffre wird aus diesem Ergebnis eine neue Matrix aufgebaut und erneut permutiert.

P	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	k_1^{-1}	$k_1^{-1}+\|K\|$	...	k_2^{-1}	$k_2^{-1}+\|K\|$	...	...	k_3^{-1}	$k_3^{-1}+\|K\|$	...	...	k_4^{-1}	$k_4^{-1}+\|K\|$	$k_4^{-1}+2\|K\|$
C	2	6	10	14	3	7	11	1	5	9	13	4	8	12

Tabelle 2.1. Gegenüberstellung von Klartext und Chiffretext

2.3.3 Einwirkende Faktoren

Wir betrachten an dieser Stelle Faktoren, die auf das Ergebnis der Chiffre einwirken können. Dafür können wir in Abbildung 2.1 zwei Matrizen mit unterschiedlicher Breite betrachten - wir gehen davon aus, dass die Spaltenvertauschung bereits stattgefunden hat. Wir befinden uns also zwischen der Spaltenpermutation und dem spaltenweisen Auslesen - die Nummerierung dient hier dem Verständnis. Die Breite wird aus der Schlüssellänge erzeugt und ist im ersten Fall 4 und im zweiten 8 Zellen breit. In beiden Abbildungen wurde die 11. Zelle markiert. Durch das spaltenweise Auslesen erhalten wir zwei unterschiedliche Ergebnisse.

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

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

Abbildung 2.1: Einwirkung der Schlüssellänge auf Positionen der Klartextzeichen innerhalb der Verschlüsselung

Die Ergebnisse wurden in Tabelle 2.2 gegenüber gestellt. Hier ist deutlich zu erkennen, dass nur durch Änderung der Schlüssellänge, also durch die Breite des Würfels, sich das Ergebnis in 14 von 16 Fällen unterscheidet. Damit stellt die Würfelbreite offensichtlich einen entscheidenen Faktor dar.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
linke Matrix	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16
rechte Matrix	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16

Tabelle 2.2. Ergebnisse von unterschiedlichen Schlüssellängen

Nicht offensichtlich ist dagegen eine weitere Abhängigkeit, die nun genauer betrachtet werden muss. So ergeben sich bei unvollständigen Würfeln bzw. Rechtecken,

also im Fall, dass die Länge des Schlüssels kein ganzzahliger Teiler der Textlänge ist, weitere Verschiebungen. Betrachten wir hierzu die Abbildung 2.2: Die Text- und Schlüssellänge ist in allen 5 Fällen identisch. Bei zeilenweiser Durchnummerierung ist in allen fünf Fällen die 7. Zelle markiert. Bei spaltenweiser Auslesung erhalten wir in den ersten drei Fällen den Inhalt dieser Zelle an der neunten Position zurück. Im vierten Fall jedoch, würde sich diese an der 8. Position befinden und im letzten Fall an der 10. Die Höhe der Spalte (hier ob drei oder vier Zellen) ist von der Schlüsselposition und der Klartextlänge abhängig. Bei einem vollständig gefüllten Würfel fände diese Unterscheidung nicht statt.

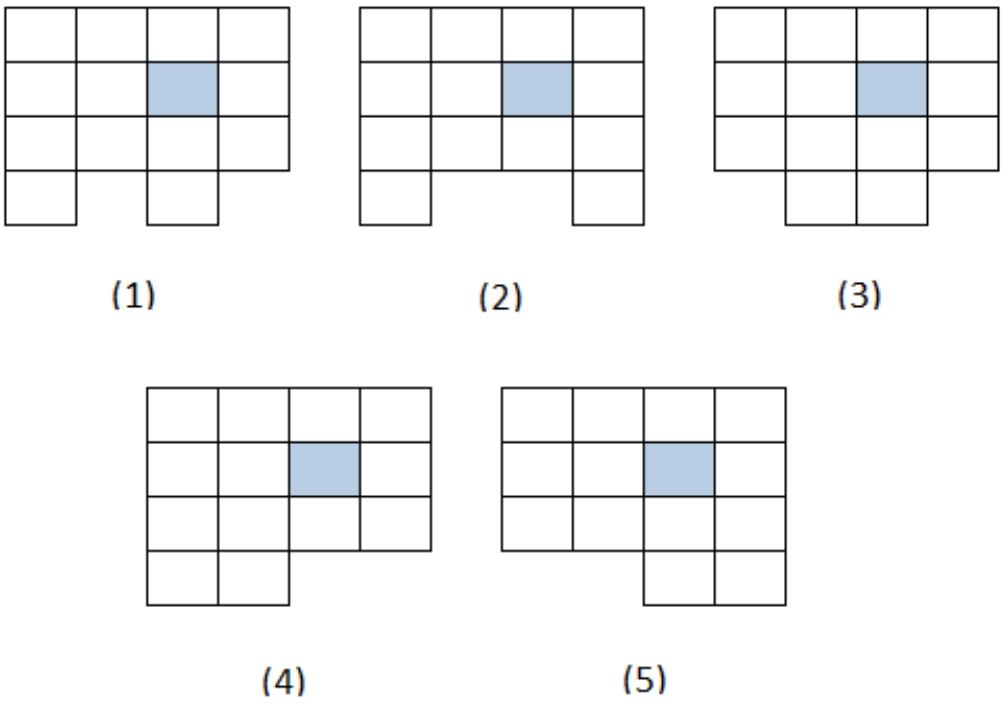


Abbildung 2.2: Unterscheidung bei unvollständiger Würfelfüllung

Es kann also festgehalten werden, dass sich nicht nur eine Schlüsselposition auf eine Spalte auswirkt, sondern auch die vorherigen Schlüsselpositionen, sofern die Matrix nicht vollständig gefüllt ist. Zusammenfassend können wir feststellen, dass folgende Faktoren auf das Ergebnis der Verschlüsselung Einfluss haben:

- die Schlüsselstelle, welche für die Spalte verantwortlich ist;
- die Länge des Schlüssels;
- die Länge des Klartextes;
- die vorherigen Schlüsselstellen.

Dies ist durchaus bemerkenswert, da in modernen Chiffren eine feste Blocklänge verwendet wird und die vorherigen Blöcke keinen direkten Einfluss auf die folgenden haben, sofern diese Abhängigkeit nicht explizit mit einem Betriebsmodus (z.B. Cipher Block Chaining) hergestellt wird.

2.3.4 Aufstellen der Chiffrierfunktion

In Abschnitt 2.2 wurde beschrieben, dass der Plaintext P über eine Funktion f nach C übertragen werden kann. Die Funktion f ordnet dabei jeder Position im Plaintext eine Position im Chiffretext zu. Offen blieb bisher die Frage, ob und wie dies innerhalb der einfachen Spaltentranspositionschiffre möglich ist. Diese Permutationsfunktion φ soll in diesem Abschnitt definiert werden und so lassen sich die Positionen des Chiffretextes als Funktion der Zahlen $1\,2\,3\,\ldots\,s$ darstellen, wobei s hier der Länge des Klartextes entspricht.

$$\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & \ldots & s \\ \varphi(1) & \varphi(2) & \varphi(3) & \varphi(4) & \varphi(5) & \ldots & \varphi(s) \end{array} \tag{2.34}$$

Aufgrund der beeinflussenden Faktoren, die im vorherigen Abschnitt dargelegt wurden, ist φ allerdings nicht von trivialer Natur. Eine reine Transposition der Matrix bzw. des Würfels nebst Spaltenvertauschung kann nicht funktionieren, da der Würfel nicht stets vollständig gefüllt ist. Wir wissen, dass die vorherigen Schlüsselpositionen hierbei eine Rolle spielen. Dieser nicht ganz intuitive Aspekt soll nun näher betrachtet werden, weswegen wir uns den in Abbildung 2.3 dargestellten Fall ansehen. Je nachdem wie die Spalten transponiert werden, ergeben sich wie auch in Abbildung 2.2 gezeigt, unterschiedliche Formen. An dieser Stelle wurde jedoch der Einfluss besonders deutlich gemacht. Die 8. Position wurde hier durch ein Rechteck markiert. Je nach Inhalt von k_1^{-1}, ist die Länge der ersten Spalte entweder 4 oder 3. Falls ersteres zutrifft, befindet sich (bei spaltenweisem Auslesen) die 8. Position im Einflussbereich von k_2^{-1}. Sofern die Länge allerdings 3 beträgt, so wird die 8. Position über k_3^{-1} berechnet. Somit gilt im ersten Fall $\varphi(8) = k_2^{-1} + 3\,|K|$ und im zweiten Fall $\varphi(8) = k_3^{-1}$.

Diese Problematik muss nun genauer untersucht werden. Wir betrachten das Problem an einem konkreten Beispiel. Erneut verwenden wir einen Plaintext der Länge $|P| = 14$ und setzen $K = \big(3\,1\,2\,4\big)$. Da $|K| = 4$, haben wir einen Rest von $r = 2$. Das bedeutet, die zweite und dritte Spalte besitzen die Länge 4 und die erste und letzte Spalte die Länge 3. Der initiale Würfel ähnelt dem dritten Beispiel in Abbildung 2.2.

Wir berechnen nun $K^{-1} = \big(2\,3\,1\,4\big)$. Die Spalte 3 ist nun also an Position 1, die Spalte 1 an Position 2, die Spalte 2 an Position 3 und die Spalte 4 an Position 4. Mit dieser Information lasen sich nun exakt die Spaltenlängen bestimmen.

An den Stellen, an denen K^{-1} einen Wert $< r$ aufweist, liegt ein Rechteck der Länge m vor, andernfalls $m - 1$. Die Abmessungen des Würfels sind in Abbildung 2.4 dargestellt.

Bei der Berechnung von φ muss also zunächst festgestellt werden, welche Spaltenlänge die jeweiligen Schlüsselpositionen erzeugen, ähnlich Abbildung 2.3.

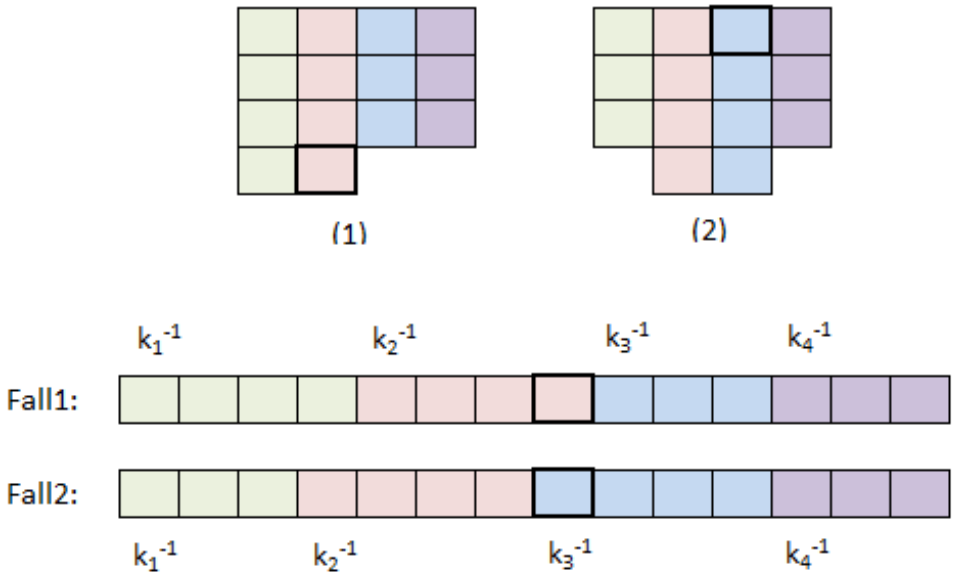


Abbildung 2.3: Einfluss von vorherigen Schlüsselpositionen

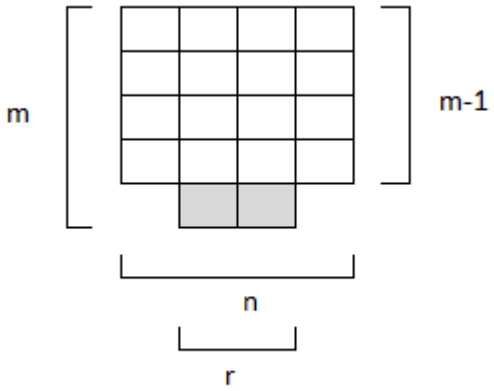


Abbildung 2.4: Die Abmessungen eines unvollständigen Würfels

Anschließend kann an den jeweiligen Startpunkten das entsprechende k_i^{-1} eingesetzt werden, wie in Tabelle 2.1 zu sehen war. Soll beispielsweise im oberen Beispiel $\varphi(11)$ bestimmt werden, so erzeugen wir über K bzw. K^{-1} die Rechtecke der Länge 3-4-4-3 und erkennen, dass sich die 11 im Einflussbereich von k_3^{-1} befindet und zwar an vierter Position. Daher gilt

$$\varphi(11) = k_3^{-1} + 3\,|K| = 2 + 12 = 14 \tag{2.35}$$

2.4 Übergang zur doppelten Spaltentransposition

Bisher wurde sich ausschließlich mit dem Problem der einfachen Spaltentransposition beschäftigt. Die doppelte Spaltentransposition sieht vor, zwei einfache Durchläufe hintereinander auszuführen.

Definition 2.6. *(Doppelwürfel) Der* Doppelwürfel *ist die zweifache Anwendung des Einfachwürfel nach Def. 2.4.*

Hierbei kann die Funktion φ zur Durchführung der ersten Runde und die Funktion ψ zur Ausführung der zweiten verwendet werden. Beispielsweise würde die 11. Position im Klartext nach $\psi(\varphi(11))$ im Chiffretext verschoben werden. Um die Sicherheit der Verschlüsselung zu erhöhen, werden häufig unterschiedliche Schlüssel und Schlüssellängen verwendet. Wird allerdings in beiden Runden der gleiche Schlüssel verwendet, so genügt eine doppelte Anwendung der φ Funktion.

Im Falle von unterschiedlichen Schlüsseln sind diese jedoch voneinander unabhängig. Da sich diese Betrachtungsweise nicht für die Kryptanalyse der doppelten Spaltentranspositionschiffre eignet, wird in Kapitel 3 die Permutationsfunktion für einige Spezialfälle der Verschlüsselung definiert und anschließend in Kapitel 4 auf den allgemeinen Doppelwürfel ausgeweitet. Diese Permutationsfunktion wird verwendet, um Angriffe auf die Verschlüsselung zu fahren.

In diesem Kapitel wurden Permutationen detailiert betrachtet und auch bzgl. ihrer mathematischen Basis untersucht. Wie gezeigt, werden diese in verschiedenen Kryptologischen Anwendungen verwendet. So zählt auch der Doppelwürfel zu der Familie der Permutationschiffren.

Um das Verfahren erneut zu verdeutlichen, wurde eine naive Permutationsfunktion definiert, welche die einfache Spaltentranspositionschiffre umsetzt. Dieses Vorgehen wird in den nächsten Kapiteln für die doppelte Spaltentranspositionschiffre reproduziert.

3

Kryptanalyse von Spezialfällen

In diesem Kapitel wird sich mit der Kryptanalyse von Spezialfällen des Verfahrens beschäftigt. Innerhalb dieser Kryptanalyse ist stets bekannt, dass das Verfahren so eingesetzt wurde, wie innerhalb der Einleitung beschrieben wurde. Zu den Spezialfällen zählen unter anderem der quadratische und der vollständig gefüllte Würfel. Ebenfalls kann die doppelte Verwendung eines Schlüssels (für beide Runden) als Spezialfall angesehen werden. Es wurden explizit diese Fälle gewählt, da sich hier eine Erweiterung zum allgemeinen Doppelwürfel am besten zeigen und durchführen lässt. Das Hauptergebnis stellt Satz 3.5 dar und für eilige Leser werden die Ergebnisse dieses Kapitels in Abschnitt 3.4 in komprimierter Form dargestellt.

In diesem und auch im nächsten Kapitel, wird stets von einer bekannten Schlüssellänge ausgegangen. Das bedeutet, die Länge der zwei verwendeten Schlüssel ist stets bekannt. Innerhalb der praktischen Anwendung werden üblicherweise kurze Schlüssel verwendet, da sich Sender und Empfänger diese einprägen müssen. Es ist also davon auszugehen, dass ein normaler Schlüssel zwischen 10 und 30 Zeichen besitzt - also 20 verschiedene Schlüssellängen. Bei zwei Runden, kann eine Schlüssellängensuche mit dem Aufwand $20^2 = 400$ durchgeführt werden. Ein solcher Faktor kann (z.B. aufgrund von Parallelisierung) üblicherweise ignoriert werden. Ebenfalls sollte klar sein, dass für alle Lösungen der gezeigten Gleichungen stets Ganzzahligkeitsforderung herrscht. Es handelt sich stets um Werte für Schlüssel- oder Textpositionen und diese stammen daher aus dem natürlichen Zahlenraum.

Kurz erwähnt soll der Fall sein, dass die Schlüssellänge identisch mit der Länge des Textes ist. Es gilt also $|P| = |C| = |K|$. Dabei liegt ein Würfel der Höhe 1 vor - in diesem Fall können die Schlüssel der zwei Runden gemäß Abschnitt 2.1.1 zu einer zusammengefasst werden. Es handelt sich also nur noch um eine einfache Spaltentransposition. Diese wurde in Abschnitt 2.3 bereits beschrieben. Da dies ein praxisferner Fall ist, bei dem es sich ganz offensichtlich nicht um einen Doppelwürfel handelt, muss er an dieser Stelle nicht näher betrachtet werden.

3.1 Quadratischer Würfel und identische Schlüssel

Wir wollen in diesen Abschnitt einen Spezialfall der Verschlüsselung betrachten. In diesem Fall entspricht die Länge des Klartextes dem Quadrat der Schlüssellänge. Das bedeutet, dass die Anzahl der Spalten gleich der Anzahl der Zeilen ist.

Definition 3.1. *(Quadratischer Würfel) Der* quadratische Würfel *ist ein Würfel gemäß Definition 2.3 für den gilt:*

$$|C| = |P| = |K|^2$$

Bei dieser Betrachtung wird ebenfalls in beiden Runden der gleiche Schlüssel verwendet, also $K1 = K2$.

$$\begin{pmatrix} [k_1] & [k_2] \dots [k_n] \\ 1 & \dots \dots \ n \\ n+1 & \dots \dots \ 2n \\ 2n+1 & \dots \dots \ 3n \\ \vdots & \vdots \ \ \vdots \ \ \vdots \\ (n-1)*n+1 & \dots \dots \ n^2 \end{pmatrix} \tag{3.1}$$

Wir können die Position der Elemente in dieser Matrix durch Angabe der Zeile und Spalte exakt bestimmen. Wir adressieren die Zeile mit i und die Spalte mit j. So befindet sich das Element $a_{i,j}$ mit $i = 1$ und $j = 2$ also $a_{1,2}$ in der ersten Zeile und zweiten Spalte. Durch Anwendung des Schlüssel k_j in der Spalte j, wird das Element $a_{i,j}$ um ein Δx in eine neue Spalte verschoben:

$$a_{i,j+\Delta x} \tag{3.2}$$

Anschließend wir die Matrix spaltenweise ausgelesen. Das bedeutet, die Spalten- und Zeilenangabe wird vertauscht:

$$a_{j+\Delta x,i} \tag{3.3}$$

Erneut wird dieses Element, in Abhängigkeit von k_i um ein Δy verschoben.

$$a_{j+\Delta x,i+\Delta y} \tag{3.4}$$

Im letzten Schritt wird die Matrix erneut spaltenweise ausgelesen bzw. transponiert.

$$a_{i+\Delta y,j+\Delta x} \tag{3.5}$$

Wichtig ist zu beachten, dass das Δx und Δy von dem jeweiligen Schlüssel der Spalte abhängig ist. Das bedeutet, das für jede Spalte ein anderes Δx bzw. Δy existiert. Um dies zu berücksichtigen, stellen wir nun eine Abhängigkeit mit dem Schlüssel $K = K1 = K2$ und den Δx bzw. Δy her:

$$a_{i+\Delta k_i,j+\Delta k_j} \tag{3.6}$$

Bei Δk_i bzw. Δk_j handelt es sich um einen positiven oder negativen Abstand. Wird beispielsweise die Spalte 3 durch den Schlüssel 1 zur ersten Spalte, ergibt sich ein $\Delta k_3 = -2$. Zur Kryptanalyse betrachten wir zunächst die Hauptdiagonale der Matrix für welche gilt $i = j$ und daher:

$$a_{i+\Delta k_i, i+\Delta k_i} \tag{3.7}$$

Es ist leicht einzusehen, dass die Elemente der Hauptdiagonale jeweils nur um ein Delta, abhängig von der jeweiligen Spalte, in beide Richtungen (sowohl Spalte als auch Zeile) verschoben wird. So befindet sich das erste Zeichen des Klartextes stets auf der Hauptdiagonalen. Ist dieses erste Zeichen gefunden, so befindet sich in dieser Zeile die Permutation der ersten Zeile des Klartextes.

3.1.1 Schwache Schlüssel im quadratischen Würfel

Wie bereits beschrieben, wird über die Schlüssel eine Spalten- und Zeilenvertauschung vorgenommen. Eine Zeilentransposition ist nicht in der Lage eine Spaltentransposition aufzuheben. Dies wäre nur dann möglich, wenn der Verschlüsselung eine weitere Runde hinzugefügt wird. Aus diesem Grund ist die alleinige Verwendung des neutralen Elements als Permutationsschlüssel als schwacher Schlüssel anzusehen.

3.2 Vollständiger Würfel mit identischen Schlüsseln

In diesem Abschnitt wird die Kryptanalyse eines weiteren Spezialfalls vorgenommen. Es handelt sich dabei um einen vollständig gefüllten Würfel in rechteckiger Form. Dieser Fall ist nicht so realitätsfern wie er im ersten Moment erscheinen mag: Gemäß einigen historischen Aufzeichnungen wurden häufig am Ende des Würfels Füllzeichen hinzugefügt um exakt diese Eigenschaft herzustellen. Der Einfachheit halber verwenden wir erneut zwei identische Schlüssel also $K = K1 = K2$.

Definition 3.2. *(Vollständiger Würfel) Ein vollständiger Würfel liegt vor, wenn der Klartext P ganzzahlig durch die Länge des Schlüssels K (nach Def. 2.5) teilbar ist. Es gilt:*

$$|P| = |C| = |K| * m$$

In Abschnitt 2.3.4 wurde eine naive Chiffrierfunktion für die einfache Spaltentranspositionschiffre aufgestellt, welche nach wenigen Vorberechnungen zu jeder Plaintextposition die passende Position im Chiffretext liefert. Da die Definition einer solchen Funktion für die Kryptanalyse sehr entscheidend ist, soll dies nun für diesen Spezialfall der doppelten Spaltentransposition getätigt werden.

3.2.1 Größe des Würfels

Grundsätzlich ist die Länge des verwendeten Schlüssels nicht bekannt. Jedoch kann durch die Eigenschaft der Vollständigkeit die Anzahl möglicher Schlüssellängen

reduziert werden. Durch Definition 3.2 wissen wir, dass die Anzahl der Klartextzeichen ganzzahlig durch die Schlüssellänge teilbar sein muss.

Über eine Faktorisierung von $|C|$ können die möglichen Schlüssellängen berechnet werden. Ist der Chiffretext beispielsweise 91 Zeichen lang, ist nur die Zerlegung $91 = 7 * 13$ möglich. Bei anderen Zahlen gestaltet sich die Bestimmung der Schlüssellängen zwar weniger eindeutig, jedoch in akzeptabler Weise aufwendig. So lässt sich die 40 in $2*2*2*5$ Primfaktorisieren. Mögliche Schlüssellängen sind hier: $2, 4, 5, 10, 20, 40$. Realistisch betrachtet kann von 4, 5 8 und 10 ausgegangen werden, welche nun als Möglichkeiten gleichwertig geprüft werden müssen. Nachdem $|K|$ und auch m bestimmt sind, kann der ursprüngliche Würfel aufgebaut werden.

3.2.2 Bestimmung der Permutationsfunktion

Im nächsten Schritt wird der Würfel anhand des Schlüssels permutiert. In diesem Beispiel verwenden wir den Schlüssel $K = \begin{pmatrix} 3 & 5 & 2 & 1 & 4 \end{pmatrix}$ für die Spaltentransposition. Anzumerken ist, dass die Anzahl der Elemente in einer Spalte stets gleich ist, da der Würfel vollständig gefüllt wird.

Ziel ist es nun, eine Permutationsfunktion zu finden, welche die doppelte Spaltentransposition unter Berücksichtigung dieses Spezialfalls leistet. Gesucht ist also eine Funktion φ, welche für die Klartextposition i und Schlüssels K die entsprechende Chiffretextposition j liefert:

$$\varphi(i, K) = j \tag{3.8}$$

wobei $\varphi : [1, |P|] \times \begin{pmatrix} k1_1 & k1_2 & \dots & k1_{|K1|} \end{pmatrix} \rightarrow [1, |C|]$.

Da es sich um eine Permutationschiffre nach Definition 2.2 handelt, gilt $|P| = |C|$. Dafür muss der Ablauf der Verschlüsselung genau betrachtet werden. Der Verschlüsselungsvorgang umfasst - wie üblich - zwei Runden. Im vorherigen Abschnitt konnte der Schlüssel zur Zeilen- und Spaltentransposition verwendet werden. Dies ist hier nicht mehr möglich, da die Elemente in einer Spalte ungleich der Anzahl in der Reihe sein können.

Wie in Abbildung 3.1 erkennbar, werden in der ersten Runde die Spalten verschoben und anschließend spaltenweise ausgelesen. Das Ergebnis ist der Ausgangswürfel für die zweite Runde. Hier ist farblich markiert, wie sich das spaltenweise Auslesen bemerkbar macht, was zugleich deutlich den Unterschied zum quadratischen Würfel aufzeigt.

So wird hier die erste Zelle zunächst an die dritte Position verschoben und anschließend durch das Auslesen an die 2*8+1-ste Position gebracht. Es ist zu erkennen, dass nach der ersten Runde, abhängig vom jeweiligen Schlüssel, 8, 16, 24 oder 32 Stellen vorangehen, bevor die Zelle erneut scheint. Genauer: befindet sich unsere Zelle in Spalte s und Zeile z, so wird sie durch den Schlüssel bzw. durch die Transposition an die Position

$$(k_s - 1) * m + z \tag{3.9}$$

verschoben. Das m gibt die Höhe des Würfels, also die Anzahl der Elemente in einer Spalte an und ist in diesem Fall 8. Die Spalte s und Zeile z lassen sich aus

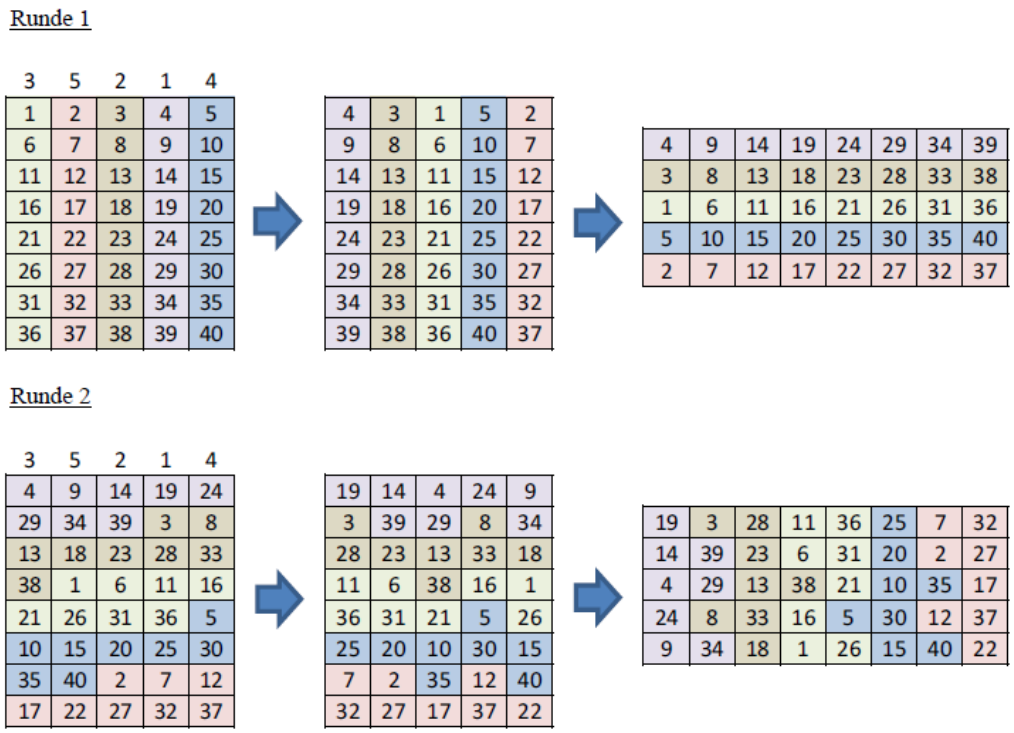


Abbildung 3.1: Verschlüsselung eines vollständig gefüllten Würfels

dem Index i des Feldes und der Schlüssellänge berechnen. Es wird also so häufig die Schlüssellänge abgezogen, bis sich das Ergebnis in $1 \dots |K|$ befindet. Da aus dem Index i also z und s berechnet werden können, werden wir diese also unter der Voraussetzung eines bekannten i und $|K|$ nun berechnen.

Satz 3.3. *Mit bekannter Schlüssellänge $|K|$ ist aus dem Index i innerhalb des Klartextes, die Zeile z und Spalte s eindeutig bestimmbar:*

$$(z-1) * |K| + s = i$$

wobei $s \in [1, |K|]$ und $z \in [1, m]$ und m gemäß Definition 3.2.

Auf diese Weise können wir nun berechnen, wie sich ein Feld nach der ersten Runde verschiebt:

$$\begin{aligned} 1 &\to (k_1 - 1) * 8 + 1 = (3-1) * 8 + 1 = 17 \\ 2 &\to (k_2 - 1) * 8 + 1 = (5-1) * 8 + 1 = 33 \\ 3 &\to (k_3 - 1) * 8 + 1 = (2-1) * 8 + 1 = 9 \\ &\dots \end{aligned} \tag{3.10}$$

Dies kann mit einem Blick auf Abbildung 3.1 bestätigt werden.

Wir hatten bereits angesprochen, dass die Tiefe des Würfels i.d.R. höher ist als breit, wodurch es beim spaltenweisen Auslesen zu Überlappungen kommt. Daher ist eine direkte Zuordnung zwischen Zelle und Schlüsselstelle im zweiten Schlüssel nicht möglich. Vielmehr ergibt sich aus dem Inhalt des ersten Schlüssels,

welcher Teil des zweiten Schlüssels verwendet werden muss. Betrachten wir erneut Abbildung 3.1. Hier ist zu sehen, dass die 1 aus der ersten Spalte nach der ersten Runde in die dritte Spalte fällt. Die 2 hingegen wird in die fünfte Spalte übertragen. Die 6 aus der ersten Spalte wird ebenfalls in die dritte Spalte verschoben. So ist zunächst von K abhängig in welche Spalte der Klartext übertragen wird. Ohne Kenntnis des ersten Schlüssels sind maximal Schätzungen, jedoch keine genauen Aussagen über die Position nach der ersten Runde möglich. Ungeachtet dessen, können wir durch Einfügen geeigneter Variablen die zweite Runde formal durchführen. Es wurde gezeigt, dass die erste Zelle durch die Schlüsselstelle k_1 an den Platz 17 verrückt wird. Daraus ermitteln wir durch Subtraktion von $|K|$ die neue Spalte:

$$17 - 3 * 5 = 2 \tag{3.11}$$

Die Zelle 1 aus der ersten Runde befindet sich also in Spalte 2 der zweiten Runde. Verallgemeinert lässt sich also sagen:

$$i' - t * |K| = s' \tag{3.12}$$

Der neue Index i' (Position nach der ersten Runde) befindet sich in Spalte s'. Das i' wurde unter Verwendung von K berechnet. Setzen wir also nun für i' die oben genannte Formel ein, so erhalten wir die folgende Gleichung:

$$(k_s - 1) * m + z - t * |K| = s' \tag{3.13}$$

Damit ist nun die Spalte innerhalb der zweiten Runde bekannt, auf die der Schlüssel K angewendet wird. Wie wir wissen, ist der Ablauf der zweiten Runde identisch mit dem der ersten. Aus diesem Grund gilt:

$$(k_{s'} - 1) * m + z' = i'' \tag{3.14}$$

Setzen wir nun für das s' den oberen Term ein, so erhalten wir:

$$(k_{(k_s-1)*m+z-t*|K|} - 1) * m + z' = i'' \tag{3.15}$$

Zuletzt muss noch z' genauer bestimmt werden. Innerhalb der ersten Runde haben wir $t * |K|$ subtrahiert um die Spalte zu ermitteln. Um einen Zusammenhang zwischen z' und t zu zeigen, betrachten wir erneut die erste Zelle, welche in Runde 1 auf die 17 verschoben wird. Die Anzahl vorheriger Zellen in der Spalte ist $t = 3$ bzw. die 1 befindet sich an der 4. Position der Spalte. In Runde 2 wurde nun die Spalte vertauscht, wodurch sich allerdings an t nichts ändert. Die Zelle befindet sich also immernoch an der vierten Position der Spalte. Durch das spaltenweise Auslesen werden nun 4 Spalten mit 8 Zellen ausgelesen zzgl. den 3 Feldern um dann im 4. Feld der Spalte auf die 17 zu treffen.

Satz 3.4. *Mit bekanntem $|K2|$ kann aus dem Ergebnis i' der ersten Runde die Zeile z' und Spalte s' der zweiten Runde gemäß Satz 3.3 berechnet werden. Für den Wert t gilt $t = (z' - 1)$ wobei $t \in [0, m_2 - 1]$.*

Nach Satz 3.4 gilt daher:

$$z' = t + 1 \tag{3.16}$$

und wir können die Formel so zusammenfassen:

$$(k_{(k_s-1)*m+z-t*|K|} - 1) * m + (t + 1) = i'' \tag{3.17}$$

Das i'' ist j. Am Beispiel aus Abbildung 3.1 können wir z.B. die 17 in Spalte $s = 2$ und Zeile $z = 4$ mittels Kenntnis von $K = \begin{pmatrix} 3 & 5 & 2 & 1 & 4 \end{pmatrix}$ verschlüsseln:

$$\begin{gathered} (k_{(k_s-1)*m+z-t*|K|} - 1) * m + (t + 1) = i'' \\ (k_{(k_2-1)*m+4-t*5} - 1) * m + (t + 1) = i'' \\ (k_{4*8+4-t*5} - 1) * 8 + (t + 1) = i'' \\ (k_{4*8+4-7*5} - 1) * 8 + (7 + 1) = i'' \\ (k_1 - 1) * 8 + 8 = i'' \\ 2 * 8 + 8 = 24 \end{gathered} \tag{3.18}$$

In diesem Beispiel ist zunächst der Schlüssel k_2 und anschließend der Schlüssel k_1 zum Einsatz gekommen. Damit wurde indirekt eine Definition für die Permutationsfunktion in 3.8 erbracht.

3.2.3 Chosen-Plaintext Angriff

Die erworbenen Kenntnisse aus dem vorherigen Abschnitt sollen nun zur Kryptanalyse der Verschlüsselung genutzt werden. Wir gehen an dieser Stelle von einem Chosen oder Known-Plaintext Angriff aus, in welchem wir die Positionspaare zwischen Klar- und Chiffretext genau bestimmen konnten. Das bedeutet, wir wissen mit absoluter Sicherheit, dass z.B. das Feld 26 auf das Feld 37 abgebildet wurde. Ebenfalls kennen wir die Länge des Schlüssels $|K| = 5$ und die Länge des Klar- bzw. Chiffretextes $|P| = |C| = 40$. Außerdem gilt immernoch, dass die Schlüssel identisch sind. Zunächst bestimmen wir über Satz 3.3 die Zeile z und Spalte s des Feldes 26 im ursprünglichen Würfel und erhalten das Ergebnis

$$(6 - 1) * 5 + 1 = i \tag{3.19}$$

also $z = 6$ und $s = 1$. Aus diesem Grund gilt:

$$(k_{(k_1-1)*m+6-t*|K|} - 1) * m + (t + 1) = 37 \tag{3.20}$$

Da $m = \frac{40}{5} = 8$ und $|K| = 5$ gilt, können wir diese ebenfalls in die Gleichung einsetzen:

$$(k_{(k_1-1)*8+6-t*5} - 1) * 8 + (t + 1) = 37 \tag{3.21}$$

Im ersten Schritt vereinfachen wir den linken Teil durch Einsatz einer Variable v, um die Bestimmung von t zu erleichtern:

$$v * 8 + (t + 1) = 37 \tag{3.22}$$

Wir betrachten nun alle ganzzahligen Lösungen von v, bei denen $t \in [0, 7]$

$$4 * 8 + (4 + 1) = 37 \tag{3.23}$$

ist die einzige Lösung und daher wissen wir das $t = 4$ und

$$(k_{(k_1-1)*8+6-4*5} - 1) = v = 4 \tag{3.24}$$

Im nächsten Schritt wird v näher betrachtet bzw. vereinfacht:

$$(k_{(k_1-1)*8-14} - 1) = 4 \tag{3.25}$$

Auch hier wollen wir erneut durch den Einsatz einer Variable die Übersichtlichkeit erhöhen und so verwenden wir v um den Index des ersten k zu beschreiben. Es gilt also $k_v - 1{=}4$. Dieses v muss allerdings bestimmt werden und daher betrachten wir dieses als nächstes:

$$(k_1 - 1) * 8 - 14 = v \tag{3.26}$$

Wir betrachten nun alle ganzzahligen Lösungen für k_1:

$$\begin{gathered} 1 * 8 - 14 = -6 \\ 2 * 8 - 14 = 2 \\ 3 * 8 - 14 = 10 \\ 4 * 8 - 14 = 18 \\ 5 * 8 - 14 = 26 \end{gathered} \tag{3.27}$$

Da allerdings $u \in [1, |K|]$ also $u \in [1, 5]$, kann nur die zweite Lösung die richtige sein. So ist $(k_1 - 1) = 2$ also $k_1 = 3$. An dieser Stelle ist daher bekannt, dass $k_u = 5$ und $k_1 = 3$ und es gilt:

$$(k_1 - 1) * 8 - 14 = v = 2 * 8 - 14 = 2 \tag{3.28}$$

womit die Kryptanalyse mit dem Ergebnis $k_1 = 3$ und $k_2 = 5$ abgeschlossen ist. Gefunden wurde also

$$K = \begin{pmatrix} 3 & 5 & 0 & 0 & 0 \end{pmatrix} \tag{3.29}$$

wobei die 0 für unbekannt steht. In diesem Beispiel wurde der Würfel aus Abbildung 3.1 einer Analyse unterzogen, welche dort auch nachvollzogen werden kann. Dabei wurden aus einem Klar-/Chiffretextpositionspaar zwei Schlüsselstellen rekonstruiert.

3.2.4 Schwache Schlüssel im vollständigen Würfel

Ähnlich wie im quadratischen Würfel, wird über den ersten Schlüssel eine Spaltenvertauschung vorgenommen, während der zweite Schlüssel im Wesentlichen die Zeilen vertauscht. Durch die Rechteckige Form und den in Abbildung 3.1 sichtbaren Überlappungseffekt, ist dies jedoch nicht ganz so trivial wie im quadratischen Würfel. Jedoch gilt auch hier, dass die zweite Runde nicht die Vertauschung der ersten aufheben kann. Die Ausführung wird daher stets zu einer Permutation führen. Eine Ausnahme bildet auch hier das neutrale Element der Permutationen, wobei in diesem Fall zwei verschiedene Schlüssellängen verwendet werden müssen.

Gegeben sei ein Klartext P der Länge $|P|$. Wird in der ersten Runde die Permutation $\begin{pmatrix}1\,2\,3\,\ldots\,n\end{pmatrix}$ verwendet, so muss in der zweiten Runde muss die Permutation $\begin{pmatrix}1\,2\,3\,\ldots\,m\end{pmatrix}$ verwendet werden, wobei: $m = \frac{|P|}{n}$. Das m ist also die Höhe des Würfels der ersten Runde. Damit wird der Würfel 2-mal transponiert ohne das eine Permutation vorgenommen wird und so entsteht am Ende wieder der ursprüngliche Klartext P.

3.3 Vollständiger Würfel mit unabhängigen Schlüsseln

Im letzten Abschnitt wurde die Kryptanalyse für den Spezialfall $K1 = K2$ durchgeführt. Dabei wurde eine Funktion für die Permutation definiert und anschließend zur Kryptanalyse genutzt, sofern ein Klar-/Chiffretext Positionspaar vorliegt. In diesem Abschnitt wird von unabhängigen Schlüsseln bzw. Schlüssellängen ausgegangen. Somit erweitert sich die gesuchte Permutationsfunktion:

$$\varphi(i, K1, K2) = j \tag{3.30}$$

wobei $\varphi : [1, |P|] \times \begin{pmatrix}k1_1\,k1_2\,\ldots\,k1_{|K1|}\end{pmatrix} \times \begin{pmatrix}k2_1\,k2_2\,\ldots\,k2_{|K2|}\end{pmatrix} \rightarrow [1, |C|]$.

Die Eigenschaft, dass der Würfel vollständig gefüllt ist, wird beibehalten. Daher müssen, wie in Unterabschnitt 3.2.1 erklärt, beide Schlüssellängen ganzzahlige Teiler der Klartextlänge sein:

$$|K1| * m_1 = |K2| * m_2 = |P| = |C| \tag{3.31}$$

Erneut kann über eine Faktorisierung die Anzahl der unterschiedlichen Schlüssellängen bestimmt werden. Sei der Fall das $|P| = 40$ gegeben, so sind die sinnvollen verschiedenen Schlüssellängen:

$$\begin{array}{ll}
|K1| = 2 & |K2| = 2 \\
|K1| = 2 & |K2| = 4 \\
|K1| = 2 & |K2| = 5 \\
|K1| = 2 & |K2| = 8 \\
|K1| = 4 & |K2| = 10 \\
|K1| = 4 & |K2| = 2 \\
|K1| = 4 & |K2| = 4 \\
\vdots & \vdots
\end{array} \tag{3.32}$$

In diesem Beispiel wird $|K1| = 4$ und $|K2| = 8$ gewählt. In Abbildung 3.2 ist dieses Verfahren unter Verwendung der Schlüssel $K1 = \begin{pmatrix}3\,2\,4\,1\end{pmatrix}$ und $K2 = \begin{pmatrix}6\,2\,8\,7\,4\,1\,5\,3\end{pmatrix}$ dargestellt.

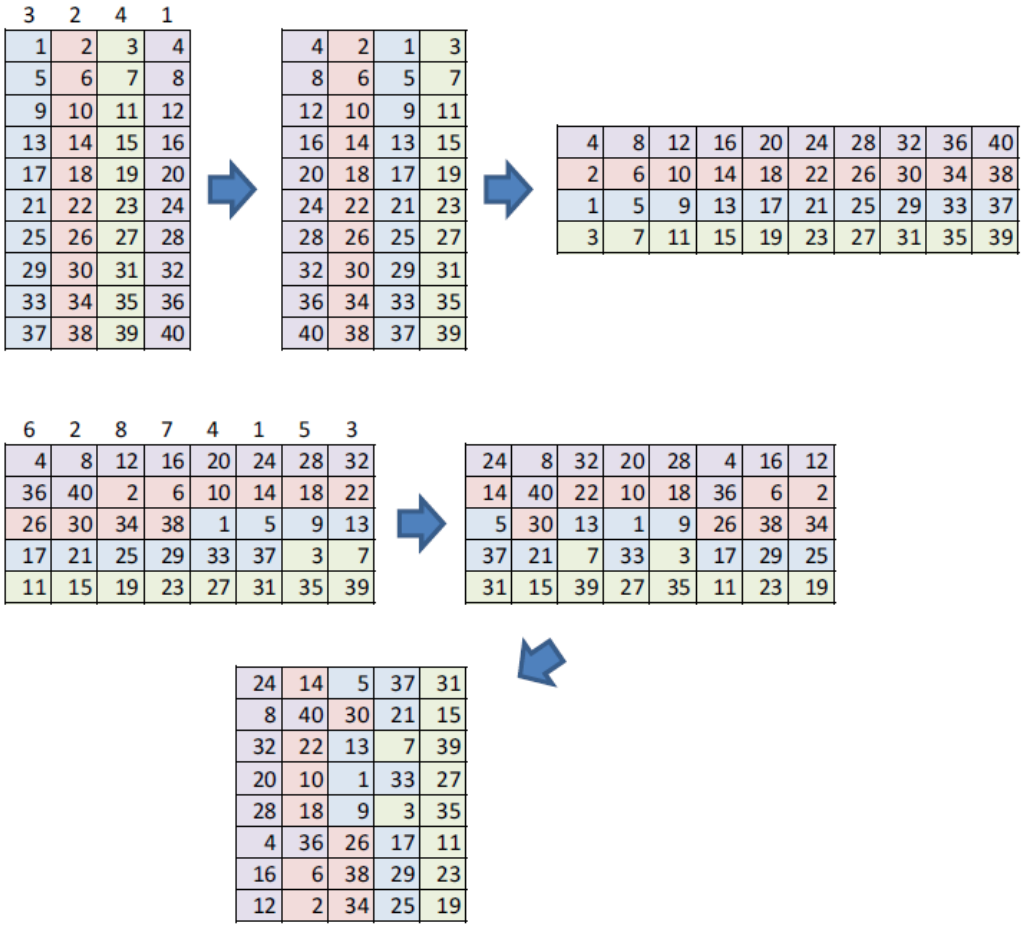


Abbildung 3.2: Verschlüsselung mit vollständiger Füllung und unabhängigen Schlüsseln

Wie leicht zu erkennen gilt $K1 \neq K2$. Ebenfalls unterscheidet sich die Tiefe des Würfels in den beiden Runden, so gilt $m_1 = 10$ und $m_2 = 5$. Im ersten Schritt verschlüsseln wir den Klartext anhand des zuständigen Spaltenschlüssels:

$$(k1_s - 1) * m_1 + z \tag{3.33}$$

Dabei sind s und z wie in Abschnitt 3.2 definiert. Das m_1 ergibt sich aus dem Verhältnis zwischen Länge des Textes und $K1$. Ist beispielsweise nach der neuen Position j der Zelle $i = 7$ gefragt, so setzen wir $s = 3$ und $z = 2$. Im nächsten Schritt folgt das spaltenweise Auslesen. Wir erinnern uns, dass diese Berechnung notwendig war, um die neue Spalte im Würfel der zweiten Runde zu bestimmen. Wir müssen daher t-oft $|K2|$ vom Ergebnis abziehen:

$$(k1_s - 1) * m_1 + z - t + |K2| = s' \tag{3.34}$$

Auf unser Beispiel mit Feld 7 bezogen:

$$\begin{gathered} (k1_3 - 1) * 10 + 2 - t * 8 = s' \\ (4 - 1) * 10 + 2 - t * 8 = s' \\ 32 - t * 8 = s' \end{gathered} \tag{3.35}$$

Da $s' \in [1, 8]$ muss gelten:

$$32 - 3 * 8 = 8 \tag{3.36}$$

So findet sich die 7 nach der ersten Runde an der Stelle 32 in Spalte 8. Für Spalte 8 ist der Schlüssel $k2_8$ zuständig. Wir erkennen an dieser Stelle, das bis auf die unterschiedlichen m-Werte und die Unterscheidung bzgl. der Schlüssel in den zwei Runden, kein Unterschied zu Abschnitt 3.2 besteht. Aus diesem Grund kann die Formel für diesen Spezialfall wie folgt definiert werden, und schließt dabei alle vorherigen Spezialfälle mit ein.

Satz 3.5. *Die Verschiebung von Klartextposition i nach Chiffretextposition j der doppelten Spaltentranspositionschiffre mit den Schlüssel $K1$ und $K2$ und vollständig gefüllten Würfeln in beiden Runden, kann über die Gleichung*

$$(k2_{(k1_s-1)*m_1+z-t*|K2|} - 1) * m_2 + (t + 1) = j$$

beschrieben werden, wobei

- *Spalte s und Zeile z nach Satz 3.3 aus i,*
- *der t-Wert nach Satz 3.4*
- *m_1 und m_2 gemäß Definition 3.2 - je nach Anwendung von $K1$ oder $K2$*

berechnet wird.

Auf unser Beispiel bezogen kann die Berechnung nun wie folgt abgeschlossen werden:

$$j = i'' = (k2_8 - 1) * 5 + (3 + 1) = 14 \tag{3.37}$$

Dieses Ergebnis lässt sich in Abbildung 3.2 nachvollziehen.

3.4 Zusammenführung der Spezialfälle

Wir konnten zeigen, dass für den Spezialfall eines ausgefüllten Rechtecks/Würfels die Berechnung der resultierenden Positionen linear vom jeweiligen Spaltenschlüssel abhängt. Falls $K1 = K2$ gilt, also ein gemeinsamer Schlüssel vorliegt, ist die Anzahl der benötigten Klar-/Chiffretextpositionspaare geringer. Sind zwei unterschiedliche Schlüssel zu suchen, so sind mehr Informationen notwendig. In allen betrachteten Fällen war die Lösung stets eindeutig und erbrachte mindestens eine Schlüsselstelle. Die Informationen aus den drei Abschnitten, sollen nun noch einmal formal zusammengefasst werden. Für die Verschlüsselung des Feldes mit dem Index i an die Position mit dem Index j gilt:

$$K1 = \left(k1_1\ k1_2\ k1_3 \ldots k1_{|K1|}\right) \forall l \in [1, |K1|] :\ 1 \leq k1_l \leq |K1| \tag{3.38}$$

$$K2 = \left(k2_1\ k2_2\ k2_3 \ldots k2_{|K2|}\right) \forall l \in [1, |K2|] :\ 1 \leq k2_l \leq |K2| \tag{3.39}$$

$$|K1| * m_1 = |K2| * m_2 = |P| = |C| \tag{3.40}$$

Für Spalte s und Zeile z gilt (Satz 3.3):

$$(z-1) * |K_1| + s = i \tag{3.41}$$

wobei $s \in [1, |K1|]$ und $z \in [1, m_1]$ und $i \in [1, |P|]$

$$v * m_2 + (t+1) = (k2_u - 1) * m_2 + (t+1) = j \tag{3.42}$$

wobei $v \in [0, |K2| - 1]$ und $u \in [1, |K2|]$. Zusammenfassend gilt (Satz 4.6):

$$(k2_{(k1_s-1)*m_1+z-t*|K2|} - 1) * m_2 + (t+1) = j \tag{3.43}$$

wobei $t \in [0, m_2 - 1]$ (Satz 3.4) und repräsentiert die Zeile des Klartextes im Würfel der zweiten Runde.

Beispiel

Abschließend soll dies erneut an einem Klar-/Chiffretextpositionspaar aus Abbildung 3.2 gezeigt werden. Wir kennen die Schlüssellängen $|K1| = 4$ und $|K2| = 8$ und dass die $i = 27$ auf die $j = 20$ in einem Text der Länge 40 abgebildet wird. Wie diese Information erhoben werden kann, wird in Abschnitt 4.5 des nächsten Kapitels genauer erklärt.

$$m_1 = \frac{40}{4} = 10 \tag{3.44}$$

$$m_2 = \frac{40}{8} = 5 \tag{3.45}$$

Nun bestimmen wir die Zeile z und Spalte s der 27 im Ausgangswürfel:

$$i = (7-1) * 4 + 3 = 27 \tag{3.46}$$

daher gilt $z = 7$ und $s = 3$. Die Ergebnisse setzen wir wie folgt ein:

$$(k2_{(k1_3-1)*10+7-t*8} - 1) * 5 + (t+1) = 20 \tag{3.47}$$

bzw. vereinfacht:

$$v * 5 + (t+1) = 20 \tag{3.48}$$

Da $t \in [0, m_2 - 1]$ also $t \in [0, 4]$ existiert nur eine Lösung für den linken Term v:

$$3 * 5 + (4+1) = 20 \tag{3.49}$$

Also gilt $k_u - 1 = u = 3$ also $k_u = 4$. Wir betrachten nun $u = (k1_3 - 1) * 10 + 7 - 4 * 8)$ und finden hierfür nur eine Lösung:

$$(k1_3 - 1) * 10 - 25 = 3 * 10 - 25 = 5 \tag{3.50}$$

Also gilt $k1_3 - 1 = 3$ also $k1_3 = 4$ und $k2_5 = 4$. Dies ist gemäß Abbildung 3.2 korrekt.

4

Kryptanalyse des Doppelwürfels

Nachdem wir im vorherigen Kapitel den Spezialfall des vollständig gefüllten Würfels betrachtet haben, soll nun in diesem die Kryptanalyse des allgemeinen Doppelwürfels behandelt werden. Es werden dabei keine weiteren Annahmen vorgenommen; die Schlüssel können also eine unterschiedliche Länge aufweisen. Der wesentliche Unterschied besteht darin, dass der Würfel nun nicht länger vollständig mit Zeichen gefüllt ist, sondern einen Restanteil aufweisen kann. Dieses Reststück ist abhängig von der Länge des Klartextes und der verwendeten Schlüssellänge der jeweiligen Runde. Zunächst soll der Unterschied zwischen einem vollständigen und einem unvollständigen Würfel geklärt werden. Nach Defintion 4.1 liegt dann ein unvollständiger Würfel vor, wenn sich in der letzen Zeile des Würfels ungenutzte Felder befinden.

Definition 4.1. *(Unvollständiger Würfel) Ein* unvollständiger Würfel *ist ein Würfel gemäß Definition 2.3, dessen Spaltenlängen sich maximal um die Länge 1 unterscheiden.*

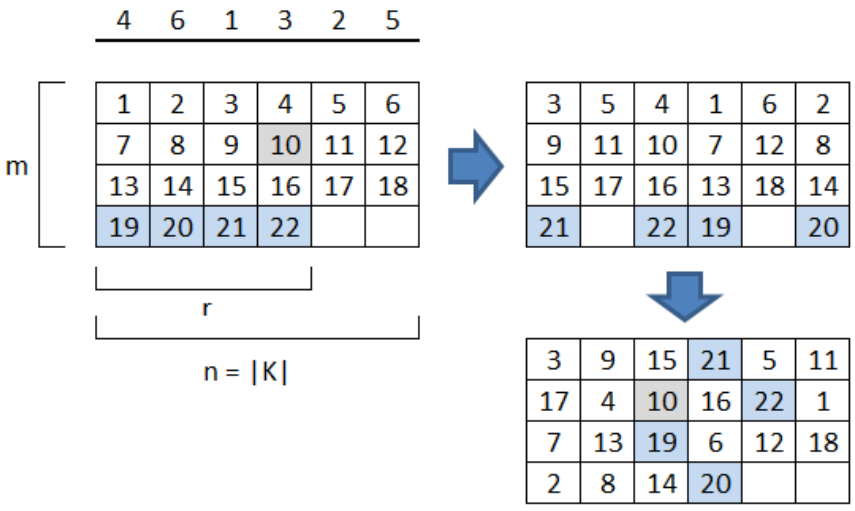


Abbildung 4.1: Spaltentransposition in der ersten Runde des Doppelwürfels mit Klartextlänge 22 und Schlüssellänge 6

In Abbildung 4.1 wurde der Schlüssel $K1 = \begin{pmatrix} 4\,6\,1\,3\,2\,5 \end{pmatrix}$ verwendet. Aus den Längen $|K| = 6$ und $|P| = 22$ kann berechnet werden, dass das Reststück eine Größe von $r = 4$ besitzt. Das bedeutet, dass die ersten vier Spalten Überlänge besitzen bzw. m-lang sind, während die zwei hinteren nur $(m-1)$-lang sind. Bisher war der m-Wert nur für den vollständigen Würfel definiert, dies soll nun für den allgemeinen Fall nachgeholt werden.

Satz 4.2. *Die Höhe m des unvollständigen Würfels kann über die Länge des Klartext P und die Länge des Schlüssels K berechnet werden:*

$$m = \left\lceil \frac{|P|}{|K|} \right\rceil$$

Auch wenn der r-Wert sofort aus der Abbildung abgelesen werden kann, soll an dieser Stelle eine Definition erfolgen.

Satz 4.3. *Für den Restwert r, den Klartext P und Schlüssel K gilt der Zusammenhang:*

$$|P| \equiv_{|K|} r$$

Auch wenn die Eigenschaft des unvollständigen Würfels zunächst unproblematisch erscheint, so wird sich gleich zeigen, dass aus diesen Restwerten innerhalb der doppelten Spaltentranspositionschiffre eine erhebliche Komplexität entsteht. Diese Problematik wurde bereits in Unterabschnitt 2.3.3 angesprochen. In Abschnitt 4.1 soll die Formel aus Satz 3.5 diesen neuen Umständen angepasst werden. Hierbei werden nur wenige Erweiterungen notwendig sein - dem Leser wird jedoch erneut ein Gesamtüberblick geboten was ggf. eine Wiederholung zu Abschnitt 3.4 aus dem vorherigen Kapitel darstellt. Nachdem die Gleichung für den allgemeinen Doppelwürfel gefunden ist, wird diese in Abschnitt 4.2 für die Kryptanalyse eingesetzt. So werden gültige Belegungen der Variablen gesucht um auf den verwendeten Schlüssel rückschließen zu können falls ein Klar-/Chiffretext Positionspaar gegeben ist. Aufgrund der Erweiterung der Formel, wird hier ein größerer Lösungsraum entstehen. Aus diesem Grund ist es notwendig, Reduktionsmöglichkeiten zu definieren um die Anzahl der möglichen Schlüssel einzuschränken. Mit diesen Reduktionen beschäftigen sich die Abschnitte 4.3 und 4.4. Da in der Praxis das Vorliegen von Positionspaaren jedoch unwahrscheinlich ist und dies eher einem Chosen-Plaintext Angriff entspricht, wird dieser in Abschnitt 4.5 zu einem Known-Plaintext ausgeweitet.

4.1 Bestimmung der Permutationsfunktion

In diesem Abschnitt soll die Permutationsfunktion φ für den allgemeinen Doppelwürfel bestimmt werden, wie es im vorherigen Kapitel für den vollständigen Doppelwürfel getätigt wurde. Im Fall des vollständig gefüllten Würfels aus Kapitel 3 wurde das Permutieren und spaltenweise Auslesen über $(k_s - 1) * m + z$ umgesetzt, wobei s die Spalte und z die Zeile des Feldes im Klartextwürfel darstellt. Das multiplizieren mit

m war uns dort möglich, da alle Spalten des Würfels die gleiche Länge aufwiesen. Im allgemeinen Doppelwürfel ist dies nicht gegeben. Deshalb gehen wir stets davon aus, dass alle Spalten keine Überlänge besitzen, also $(m-1)$-lang sind. Den Rest, also die Spalten die Überlänge besitzen, kompensieren wir innerhalb der ersten Runde über eine Variable a. Diese Variable a beinhaltet die Anzahl der vorherigen überlangen Spalten nach Anwendung von $K1$.

Wir betrachten den zweiten Würfel (oben rechts) in Abbildung 4.1 um dies nachvollziehen zu können. Nach der Anwendung des Schlüssels $K1 = \begin{pmatrix} 4\,6\,1\,3\,2\,5 \end{pmatrix}$ wurden die vier überlangen Spalten des Klartextwürfels neu verteilt. Der erste Schlüsselwert ist 4, was bedeutet, die erste Spalte wurde zur vierten Spalte verschoben. Wir wollen nun den a-Wert, also die Anzahl der überlangen Spalten, für diesen Schlüsselwert bestimmen. Wir sehen in der Abbildung, das vor der vierten Spalte zwei überlange Spalten zu finden sind. Das bedeutet, der korrespondierende a-Wert zum ersten Spaltenschlüssel mit dem Wert 4 ist 2.

$$a_1 = 2 \tag{4.1}$$

Nun wollen wir a_2 bestimmen. Die zweite Spalte wird über K an das Ende des Würfels verschoben. Das bedeutet, vor dieser befinden sich drei überlange Spalten und damit gilt:

$$a_2 = 3 \tag{4.2}$$

Dieses Vorgehen kann so fortgeführt werden um die Werte für $a = \begin{pmatrix} a_1\ a_2 \ldots a_n \end{pmatrix}$ nacheinander zu bestimmen, wobei $n = |K|$ gilt.

Satz 4.4. *Bekannt sei der Schlüssel $K1$ der Länge n gemäß Def. 2.5. Die Berechnung von $a = \begin{pmatrix} a_1\ a_2 \ldots a_n \end{pmatrix}$ an der Stelle s erfolgt über:*

$$a_s = \sum_{i=1}^{r} \begin{cases} 1 & falls\ k1_i < s \\ 0 & sonst \end{cases}$$

Der Wert r wird gemäß Satz 4.3 berechnet.

Im Beispiel aus Abbildung 4.1 gilt $r = 4$, $K1 = \begin{pmatrix} 4\,6\,1\,3\,2\,5 \end{pmatrix}$ und a ist nach Satz 4.4 definiert als $a = \begin{pmatrix} 2\,3\,0\,1\,1\,3 \end{pmatrix}$. Aus Gründen der Übersicht wird in dieser Arbeit das a häufig ohne Index angegeben - gemeint ist in diesen Fällen stets das korrespondierende a zum verwendeten Spaltenschlüssel. Wird also der Spaltenschlüssel k_4 verwendet, so ist mit a dementsprechend a_4 gemeint.

Wir erweitern nun die Gleichung 3.9 aus dem vorherigen Kapitel, die in Satz 3.5 verwendet wurde. Wir beschränken uns zunächst also auf die ersten Runde der Verschlüsselung und wollen den oben beschriebenen a-Wert integrieren. Dabei gehen wir, wie erwähnt, stets von $(m-1)$ langen Spalten aus und fügen dafür die Variable a hinzu:

$$C = (k_s - 1) * (m - 1) + a + z \tag{4.3}$$

Das C aus 4.3 ist die Position des Klartextzeichens nach Ausführung der ersten Runde. Im Beispiel aus Abbildung 4.1 befindet sich das Feld $P = 10$ in der Spalte

$s = 4$ und Zeile $z = 2$. Durch Kenntnis von $K1$ wissen wir, dass die Spalte $s = 4$ durch k_4 nach Spalte 3 verschoben wird.

$$\begin{aligned} C &= (3-1) * (4-1) + a + 2 \\ C &= 2 * 3 + a + 2 \\ C &= 8 + a \end{aligned} \tag{4.4}$$

Wir wissen nun, das die Position 10 auf $8 + a$ verschoben wird. Mit Anwendung von Satz 4.4 wissen wir sofort, dass der korrespondierene a-Wert zu k_4 den Wert 1 besitzt. Jedoch wollen wir dies erneut nachvollziehen um die Bedeutung des a-Wertes nochmals verdeutlichen zu können. So kann sich a nur zwischen 0 und r befinden, also in $[0, 4]$. Aus diesem Grund können nur $C = 8$, $C = 9$, $C = 11$ und $C = 12$ in Frage kommen. Da wir den Schlüssel kennen wissen wir, dass die dritte Spalte (überlang) des Klartextwürfels auf die erste Spalte verschoben wurde und die fünfte Spalte (nicht überlang) auf die zweite Position. Daher befindet sich vor Spalte 3 nur eine Spalte mit Überlänge und es gilt $a = 1$:

$$C = 8 + 1 = 9 \tag{4.5}$$

Wir haben nun erfolgreich die erste Runde der Verschlüsselung berechnet. Die zweite Runde erfolgt analog über das Einfügen einer Variable b. Diese kompensiert, ähnlich wie a, innerhalb der zweiten Runde das entstandende Reststück. Da im Doppelwürfel die zwei Runden vom Ablauf identisch sind und sich nur durch den verwendeten Schlüssel unterscheiden, werden a und b-Wert auf ähnliche Weise ermittelt.

Satz 4.5. *Es liegt der Schlüssel* $K2 = \left(k2_1\ k2_2 \ldots k2_n\right)$ *vor. Die Berechnung von* $b = \left(b_1\ b_2 \ldots b_n\right)$ *erfolt analog zu Satz 4.4 mit Anwendung von* $K2$ *anstelle* $K1$*.*

In dieser Arbeit wird häufig von einem a/b-Wert gesprochen werden. Damit ist der a-Wert gemeint, falls es sich um die erste Runde der Verschlüsselung oder der b-Wert falls es sich um die zweite Runde der Verschlüsselung handelt. Aufgrund von Satz 4.5 wissen wir, dass diese auf die gleiche Weise berechnet werden.

Satz 4.6. *Die Verschiebung von Klartextposition i nach Chiffretextposition j der doppelten Spaltentranspositionschiffre mit den Schlüsseln* $K1$ *und* $K2$ *kann über die Gleichung*

$$(k2_{(k1_s-1)*(m_1-1)+a+z-t*|K2|} - 1) * (m_2 - 1) + b + (t+1) = j$$

beschrieben werden, wobei

- *Spalte s und Zeile z nach Satz 3.3 aus i,*
- *der t-Wert nach Satz 3.4,*
- m_1 *und* m_2 *gemäß Satz 4.2 - je nach Anwendung von* $K1$ *oder* $K2$ *und*
- *die Werte a und b für den jeweiligen Spaltenschlüssel gemäß den Sätzen 4.4 und 4.5*

berechnet wird.

Zusammengefasst besteht der Unterschied zum vollständig gefüllten Würfel darin, dass $(m-1)$ anstelle m angenommen wird und die fehlenden Stellen über die Variablen a in der ersten und b in der zweiten Runde kompensiert werden. Sofern es sich um einen vollständig gefüllten Würfel $(r=0)$ handelt, verweise ich auf das vorherige Kapitel.

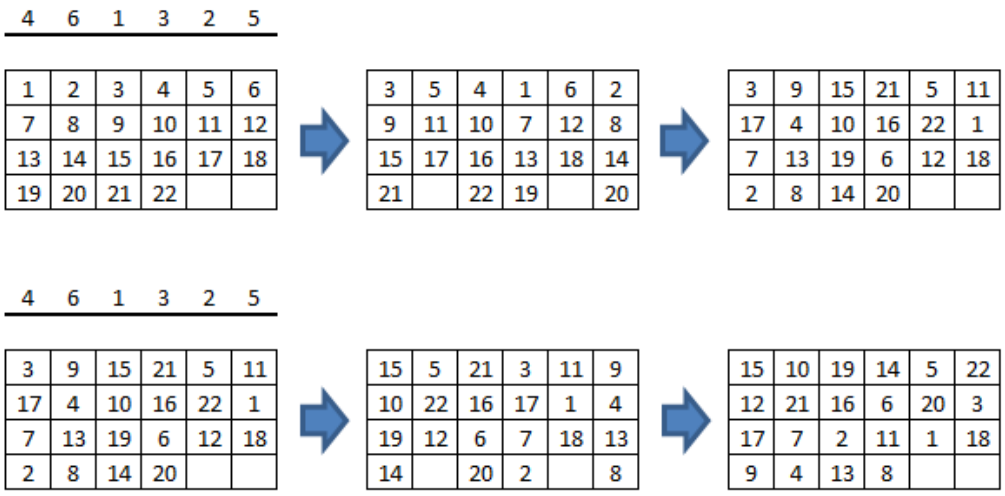


Abbildung 4.2: Durchlauf des allgemeinen Doppelwürfels mit Klartextlänge 22 und Schlüssellänge 6

In Abbildung 4.2 ist der vollständige Durchlauf des Doppelwürfels anhand der Schlüssel $K1 = \left(4\,6\,1\,3\,2\,5\right)$ und $K2 = \left(4\,6\,1\,3\,2\,5\right)$ dargestellt. O.B.d.A. wurde hier $K1 = K2$ gewählt; das Verfahren setzt dies nicht voraus und soll an dieser Stelle nur zum Verständnis beitragen bzw. das Gedächtnis des Lesers entlasten. So wollen wir nun, im ersten Schritt, die Verschiebung des 14. Feldes ermitteln, welches sich in Spalte $s=2$ und Zeile $z=3$ befindet. Dazu beginnen wir mit der Gleichung aus Satz 4.6 und tragen bekannte Werte ein:

$$\begin{aligned} C &= (k2_{(k1_s-1)*(m_1-1)+a+z-t*|K2|} - 1) * (m_2 - 1) + b + (t+1) \\ &= (k2_{(6-1)*(4-1)+a+3-t*|K2|} - 1) * (4-1) + b + (t+1) \\ &= (k2_{18+a-t*6} - 1) * 3 + b + (t+1) \end{aligned} \tag{4.6}$$

Nun ist die Bestimmung von a und b erforderlich - nachdem diese bekannt sind ergibt sich der Wert t von selbst. So kann beobachtet werden, dass der Wert a den Index von $k2$ beeinflusst. Das bedeutet, je nachdem welchen Wert a einnimmt, so verschiebt sich das Feld innerhalb der Zeile des Rundenergebnisses. Daraus folgt, dass in der zweiten Runde ein anderer Spaltenschlüssel verwendet werden müsste. Das Reststück wirkt sich also direkt auf das Resultat der ersten Runde und damit auch entscheidend auf die zweite Runde aus.

Da sich die 14 in der zweiten Spalte befindet, kommt hier $k1_2$ zum Einsatz welches die zweite Spalte zur 6. Spalte verschiebt. Das korrespondierende a zur 6.

Spalte ist $a = 3$, da sich vor der 6. Spalte, drei Spalten mit Überlänge befinden die zu kompensieren sind - vorher befinden sich drei Spalten der Länge m und zwei Spalten der Länge $(m-1)$. Mit Kenntnis von $a = 3$ kann nun folgendes festgehalten werden:

$$\begin{aligned} C &= (k2_{18+3-t*6} - 1) * 3 + b + (t+1) \\ C &= (k2_{21-3*6} - 1) * 3 + b + (3+1) \\ C &= (k2_3 - 1) * 3 + b + 4 \\ C &= (1-1) * 3 + b + 4 \\ C &= b + 4 \end{aligned} \tag{4.7}$$

Um nun C zu ermitteln muss also zunächst b bekannt sein. Bei b handelt es sich um die Anzahl der Spalten mit Überlänge innerhalb der zweiten Runde der Verschlüsselung, die sich vor der resultierenden Spalte (hier 1) befinden. Da in der zweiten Runde die dritte Spalte zur ersten verschoben wird ($k2_3 = 1$), kann sich vorher keine Spalte mit Überlänge befinden die zu kompensieren wäre. Daher muss hier $b = 0$ gelten und damit:

$$C = 0 + 4 = 4 \tag{4.8}$$

Das Ergebnis kann in Abbildung 4.2 nachvollzogen werden.

Bevor wir uns im nächsten Abschnitt der Kryptanalyse widmen, wollen wir erneut alle beteiligten Variablen aus Satz 4.6 einzeln bzgl. dem Verwendungszweck, der Zusammensetzung, der Berechnung und dem gültigen Wertebereich betrachten. Dies stellt zwar teilweise eine Wiederholung zu Abschnitt 3.4 dar, wurde jedoch um die Besonderheiten des allgemeinen Doppelwürfels erweitert.

Name	$K1$
Erklärung	Die Permutation, welche als Schlüssel der ersten Runde verwendet wird.
Zusammensetzung	$K1 = \left(k1_1\ k1_2 \ldots k1_{\lvert K1\rvert}\right)$
Berechnung	-
gültiger Wertebereich	$\forall l \in [1, \lvert K1\rvert] : 1 \leq k1_l \leq \lvert K1\rvert$

Name	$K2$
Erklärung	Die Permutation, welche als Schlüssel der zweiten Runde verwendet wird.
Zusammensetzung	$K1 = \left(k2_1\ k2_2 \ldots k2_{\lvert K2\rvert}\right)$
Berechnung	-
gültiger Wertebereich	$\forall l \in [1, \lvert K2\rvert] : 1 \leq k2_l \leq \lvert K2\rvert$

Name	m_1
Erklärung	Die Höhe des Würfels der ersten Runde.
Zusammensetzung	Zahl
Berechnung	$m_1 = \left\lceil \frac{\lvert P \rvert}{\lvert K1 \rvert} \right\rceil$
gültiger Wertebereich	$m_1 \in \mathbb{N}$

Name	m_2
Erklärung	Die Höhe des Würfels der zweiten Runde.
Zusammensetzung	Zahl
Berechnung	$m_2 = \left\lceil \frac{\lvert P \rvert}{\lvert K2 \rvert} \right\rceil$
gültiger Wertebereich	$m_2 \in \mathbb{N}$

Name	r
Erklärung	Die Anzahl der überlangen Spalten (Rest) im Klartextwürfel mit Klartext P und Schlüssel K
Zusammensetzung	Zahl
Berechnung	$\lvert P \rvert \equiv_{\lvert K \rvert} r$
gültiger Wertebereich	$r \in [0, \lvert K \rvert - 1]$

Name	z und s
Erklärung	Zeile und Spalte des Feldes mit Index i im Klartextwürfel der ersten Runde unter Verwendung der Länge von $K1$
Zusammensetzung	Zahlen
Berechnung	$(z-1) * \lvert K1 \rvert + s = i$
gültiger Wertebereich	$s \in [1, \lvert K1 \rvert]$, $z \in [1, m_1]$

Name	a
Erklärung	Anzahl der vorherigen überlangen Spalten nach Anwendung des Schlüssels $K1$ mit r-vielen überlangen Spalten der ersten Runde
Zusammensetzung	$a = \left(a_1 \; a_2 \; \ldots \; a_{\lvert K1 \rvert} \right)$
Berechnung	$a_s = \sum_{i=1}^{r} \begin{cases} 1 & falls \; k1_i < s \\ 0 & sonst \end{cases}$
gültiger Wertebereich	genaue Betrachtung folgt in Abschnitt 4.3

Name	b
Erklärung	Anzahl der vorherigen überlangen Spalten nach Anwendung des Schlüssels $K2$ mit r-vielen überlangen Spalten der zweiten Runde
Zusammensetzung	$b = \left(b_1\ b_2\ \ldots\ b_{\lvert K2\rvert}\right)$
Berechnung	$b_s = \sum_{i=1}^{r} \begin{cases} 1 & falls\ k2_i < s \\ 0 & sonst \end{cases}$
gültiger Wertebereich	genaue Betrachtung folgt in Abschnitt 4.3

Name	t
Erklärung	Dieser Wert gibt (indirekt) die Zeile des Klartextzeichens im Ausgangswürfel der zweiten Runde an. Dies ist zur Ermittlung der zuständigen Spalte der zweiten Runde und für das spaltenweise Auslesen am Ende des Verschlüsselungsvorgangs wichtig.
Zusammensetzung	Zahl
Berechnung	$t = (z' - 1)$
gültiger Wertebereich	$t \in [0, m_2 - 1]$

4.2 Kryptanalyse mittels Permutationsfunktion

In Satz 4.6 wurde die Gleichung für den allgemeinen Doppelwürfel aufgestellt:

$$(k2_{(k1_s-1)*(m_1-1)+a+z-t*\lvert K2\rvert} - 1) * (m_2 - 1) + b + (t + 1) \tag{4.9}$$

Ähnlich wie in Kapitel 3 soll sie für die Kryptanalyse eingesetzt werden. Sind die zwei Schlüssel bekannt, so lässt sich mit dieser Formel die Position eines Klartextzeichens im Chiffretext berechnen. Sind die Schlüssel nicht bekannt, Verschiebungen zwischen Klar- und Chiffretext aber schon, so soll versucht werden gültige Belegungen der Variablen zu finden um damit Schlüsselstellen herauszufinden.

Vereinfachen wir die obige Formel indem wir den linken Faktor durch v ersetzen, so erhalten wir:

$$v * (m_2 - 1) + b + (t + 1) \tag{4.10}$$

Somit muss $v \in [0, |K2| - 1]$ gelten. Greifen wir das Beispiel aus dem vorherigen Abschnitt (Verschiebung des 14. Feldes an die 4. Stelle) erneut auf, so können wir an dieser Stelle die folgenden Werte eintragen:

$$v * 3 + b + (t + 1) = 4 \tag{4.11}$$

Die Anzahl der möglichen Lösungen für diese Gleichung ist aufgrund der oben gezeigten Einschränkungen sehr überschaubar:

$$\begin{array}{l} 1)\ 0 * 3 + 0 + (3 + 1) \\ 2)\ 0 * 3 + 1 + (2 + 1) \\ 3)\ 0 * 3 + 2 + (1 + 1) \\ 4)\ 0 * 3 + 1 + (0 + 1) \\ 5)\ 1 * 3 + 0 + (0 + 1) \end{array} \tag{4.12}$$

Um diese Lösungsmenge zu reduzieren ist es notwendig, b und t stärker einzuschränken. Dafür müssen wir uns erneut in Erinnerung rufen, wofür diese zwei Variablen stehen. So ist b die Anzahl vorheriger überlanger Spalten nach Anwendung der Permutation. Bei der oben gezeigten Ersetzung, haben wir $(k2_u - 1)$ durch v ersetzt, wobei sich u aus dem Ergebnis der ersten Runde zusammensetzt. Das bedeutet es gilt $v = k2_u - 1$. In der oben gezeigten Lösungmenge ist v entweder 0 oder 1. Das heisst, $k2_u$ ist entweder 1 oder 2. Da der ersten Spalte keine überlangen Spalten vorhergehen können und der zweiten Spalte nur eine, sind Lösungen wie $v = 0 \wedge b = 3$ unmöglich. Allgemein gilt:

$$b \leq v \tag{4.13}$$

Betrachten wir unter dieser Annahme die berechnete Lösungmenge in 4.12, so können 2, 3 und 4 als unmöglich betrachtet werden. Es bleiben übrig 1 und 5.

$$\begin{array}{l} 1)\ 0 * 3 + 0 + (3 + 1) \\ 5)\ 1 * 3 + 0 + (0 + 1) \end{array} \tag{4.14}$$

Da in allen Fällen $t \in [0, m_2 - 1]$ gilt, sind hier keine weiteren Reduzierungen möglich. Damit ist der verwendete Schlüssel für die zweite Runde bereits auf zwei mögliche Lösungen eingeschränkt: $k2_u = 1$ oder $k2_u = 2$. Wir wollen im nächsten Schritt das u berechnen. Wir wissen, das innerhalb von u das t erneut verwendet wird und betrachten also den Inhalt von v der ersten Gleichung in 4.14.

$$v = (k2_u - 1) = 0 \tag{4.15}$$

$$k2_{(k1_s-1)*(m_1-1)+a+z-t*|K2|} = 1 \tag{4.16}$$

Wir vereinfachen diesen Ausdruck erneut, indem wir nur den Index von $k2$, also u als Gleichung darstellen:

$$u = (k1_s - 1) * (m_1 - 1) + a + z - t * |K2| \tag{4.17}$$

Wir wissen, das für $k2_j$ gilt, das $j \in [1, |K2|]$, so muss hier $u \in [1, 6]$ sein. Ebenfalls wissen wir aus Gleichung 1) das $m_1 = 4$ und $t = 3$.

$$u = (k1_s - 1) * (4 - 1) + a + z - 3 * 6 \tag{4.18}$$

Da uns bekannt ist, dass die 14. Stelle im Klartext auf die 4. Stelle im Ciphertext verschoben wurde, können wir mit dieser Information die Zeile (z) und Spalte (s) im Klartextwürfel berechnen.

$$\begin{gathered} u = (k1_2 - 1) * 3 + a + 3 - 3 * 6 \\ u = (k1_2 - 1) * 3 + a - 13 \end{gathered} \tag{4.19}$$

An dieser Stelle müssen alle Lösungen für diese Gleichung gleichberechtigt betrachtet werden. Hierbei ergibt sich folgende Untermenge:

$$\begin{gathered} 1.1)\ (6-1) * 3 + 1 - 15 = 1 \\ 1.2)\ (6-1) * 3 + 2 - 15 = 2 \\ 1.3)\ (6-1) * 3 + 3 - 15 = 3 \\ 1.4)\ (6-1) * 3 + 4 - 15 = 4 \end{gathered} \tag{4.20}$$

An dieser Stelle kann gestoppt werden, da $a \leq r_1$ und $r = 4$ gilt. Die Anzahl der überlangen Spalten kann nicht größer als r_1 sein. Darüber hinaus kann in diesem Beispiel die Gleichung 1.4) in 4.20 ebenfalls gestrichen werden, da in der ersten Runde $k1_2$, also die zweite Spalte verschoben wird. Da es sich bei der zweiten Spalte um eine überlange Spalte handelt kann in diesem Fall a maximal den Wert 3 einnehmen.

Im nächsten Schritt bedienen wir uns der zweiten Gleichung aus 4.14 (also 5) und stellen analog die Gleichungen für v auf.

$$v = k2_u - 1 = 1 \tag{4.21}$$

Erneut wird das u allein betrachtet:

$$\begin{gathered} u = (k1_2 - 1) * 3 + a + 3 - 0 * 6 \\ u = (k1_2 - 1) * 3 + a + 3 \end{gathered} \tag{4.22}$$

Da $u \in [1, 6]$ und $k1_2 \in [1, 6]$, kommen hier in Betracht:

$$\begin{gathered} 5.1)\ (1-1) * 3 + 0 + 3 = 3 \\ 5.2)\ (1-1) * 3 + 1 + 3 = 4 \\ 5.3)\ (1-1) * 3 + 2 + 3 = 5 \\ 5.4)\ (1-1) * 3 + 3 + 3 = 6 \\ 5.5)\ (2-1) * 3 + 0 + 3 = 6 \end{gathered} \tag{4.23}$$

Da auch auch hier gilt, dass $a < k1_2$ kommen hier nur die erste und die letzte Lösung in Frage:

$$\begin{gathered} 5.1)\ (1-1) * 3 + 0 + 3 = 3 \\ 5.5)\ (2-1) * 3 + 0 + 3 = 6 \end{gathered} \tag{4.24}$$

Wir haben nun für alle möglichen Gleichungen der zweiten Runde (1 und 5) die Gleichungen der ersten Runde aufgestellt. Sie setzen sich aus dem verbleibenden Teil (ohne 1.4) von 4.20 und den Gleichungen aus 4.24 zusammen. Sie lauten für die Gleichungen aus 4.20:

$$\begin{gathered} (k2_{(k1_2-1)*3+1+3-3*6} - 1) * 3 + 0 + (3+1) = 4\ wobei\ k1_2 = 6\ k2_1 = 1 \\ (k2_{(k1_2-1)*3+2+3-3*6} - 1) * 3 + 0 + (3+1) = 4\ wobei\ k1_2 = 6\ k2_2 = 1 \\ (k2_{(k1_2-1)*3+3+3-3*6} - 1) * 3 + 0 + (3+1) = 4\ wobei\ k1_2 = 6\ k2_3 = 1 \end{gathered} \tag{4.25}$$

Und für die Gleichungen aus 4.24:

$$\begin{array}{l}(k2_{(k1_2-1)*3+0+3-0*6}-1)*3+0+(0+1)=4\ wobei\ k1_2=1\ k2_1=2\\(k2_{(k1_2-1)*3+0+3-0*6}-1)*3+0+(0+1)=4\ wobei\ k1_2=2\ k2_6=2\end{array} \tag{4.26}$$

Diese Ergebnisse sollen nun in einer einfachen Schreibweise zusammengefasst werden. Hierbei genügt es, sich den Schlüsselwert an der entsprechenden Schlüsselposition und den zugehörigen a/b-Wert festzuhalten. Wie schon häufig angesprochen, steht der a/b-Wert in direkter Verbindung mit dem jeweiligen Spaltenschlüssel. Aus diesem Grund bietet es sich an, eine gemeinsame Schreibweise sowohl für a/b-Wert, also auch für den jeweiligen Schlüssel zu definieren. So verbinden wir die Definition des Schlüssel mit der Definition des a/b-Wertes aus Satz 4.4 und 4.5. Wir separieren zwischen Schlüssel an erster Stelle und a/b-Wert an zweiter Stelle mit einem „|"-Zeichen. Dies ist die erweiterte Darstellung des Schlüssels um die Anzahl der vorhergehenden überlangen Spalten.

$$K1^{ext} = \left(k1_1|a_1\ k1_2|a_2\ k1_3|a_3\ \ldots\ k1_{|K1|}|a_{|K1|}\right) \tag{4.27}$$

Analog für K2:

$$K2ext = \left(k2_1|b_1\ k2_2|b_2\ k2_3|b_3\ \ldots\ k2_{|K2|}|b_{|K2|}\right) \tag{4.28}$$

Übertragen auf das oben gezeigte Beispiel wäre die entstandene Lösungsmenge aus 4.25 und 4.26 in erweiterter Form hier:

$$\begin{array}{l}\left(0\ 6|1\ 0\ 0\ 0\ 0\right)\\\quad\left(1|0\ 0\ 0\ 0\ 0\ 0\right)\\\left(0\ 6|2\ 0\ 0\ 0\ 0\right)\\\quad\left(0\ 1|0\ 0\ 0\ 0\ 0\right)\\\left(0\ 6|3\ 0\ 0\ 0\ 0\right)\\\quad\left(0\ 0\ 1|0\ 0\ 0\ 0\right)\\\left(0\ 1|0\ 0\ 0\ 0\ 0\right)\\\quad\left(0\ 0\ 2|0\ 0\ 0\ 0\right)\\\left(0\ 2|0\ 0\ 0\ 0\ 0\right)\\\quad\left(0\ 0\ 0\ 0\ 0\ 2|0\right)\end{array} \tag{4.29}$$

Die „0" zeigt an, dass der Schlüssel bzw. a/b-Wert an dieser Stelle nicht bekannt ist. Dies ist die resultierende Lösungsmenge der Kryptanalyse unseres Beispielwürfels bzgl. der Verschiebung des 14. Feldes im Klartext auf die 4. Position im Chiffretext. Wir wissen anhand Abbildung 4.2, dass die dritte Lösung die tatsächlich korrekte ist. Ohne Kenntnis des Schlüssels sind zunächst jedoch alle Lösungen als gleichberechtigt zu betrachten und zu verwerten. Im nächsten Abschnitt wird das Verhältnis zwischen Spaltenschlüssel und a/b-Wert intensiver betrachtet. Aufgrund dieser Kenntnise wird es möglich sein, die vorliegende Lösungsmenge um erneut zwei Lösungen auf drei zu reduzieren.

4.3 Verhältnis zwischen Spaltenschlüssel und a/b-Wert

An einigen Stellen wurden bereits Abschätzungen bzgl. der a/b-Werte vorgenommen, die an den jeweiligen Stellen offensichtlich waren. In diesem Abschnitt soll diese Problematik vertieft werden. Zur Erinnerung: Die a/b-Werte kompensieren die vorherigen überlangen Spalten nach der Permutation. Die Anzahl der möglichen überlangen Spalten wird durch die Länge des Klartextes und die Breite des Würfels - also die verwendete Schlüssellänge bestimmt - und mit r berechnet. Bei einem Würfel der Breite 8 und einer Klartextlänge von 29 bleibt ein Rest von $r = 5$ übrig da $3 * 8 + 5 = 29$ ist.

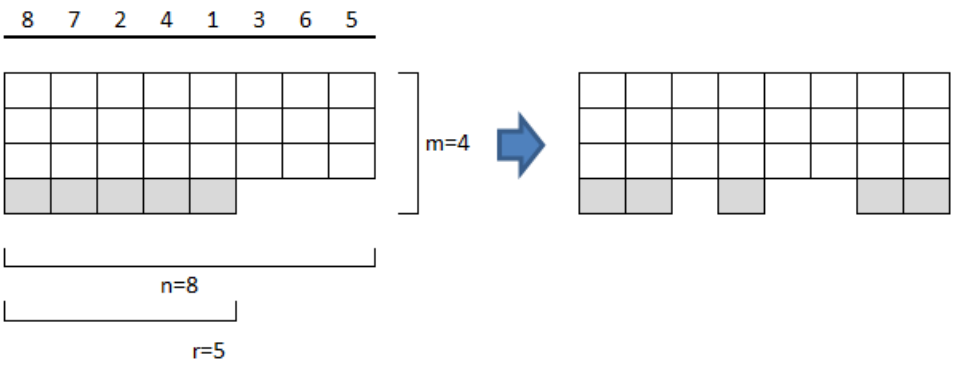


Abbildung 4.3: Würfel der Größe 29 und fünf überlangen Spalten

In Abbildung 4.3 wurde ein Würfel mit fünf überlangen Spalten dargestellt. Durch Spaltentransposition mit dem Schlüssel $K = \left(8\,7\,2\,4\,1\,3\,6\,5\right)$ werden diese überlangen Spalten neu verteilt und ergeben ein neues Gebilde. Wird wie hier beispielsweise die erste Spalte an das Ende verschoben, so befindet sich dort eine überlange Spalte. So ist ebenfalls klar, das alle vorherigen Spalten einen a/b-Wert zwischen 0 und 4 aufweisen müssen.

Wir berechnen nun die a/b-Werte der Abbildung 4.3 (Anwendung von Satz 4.4). Betrachten wir hierzu den Schlüssel K, so wird die erste Spalte auf die letzte Stelle verschoben und man erhält hierfür den a/b-Wert 4, da die letzte Spalte alle vorherigen überlangen Spalten kompensieren muss. Die zweite Spalte wird an die siebte Stelle verschoben welcher drei überlange Spalten vorausgehen. Die dritte Spalte wird an die zweite Stelle verschoben und da es sich bei der ersten Spalte um eine überlange handelt, muss die zweite Stelle diese kompensieren. Das Ergebnis dieses Vorgehens ist:

$$\left(4\,3\,1\,2\,0\,2\,3\,3\right) \tag{4.30}$$

Und in Verbindung mit dem Schlüssel $K = \left(8\,7\,2\,4\,1\,3\,6\,5\right)$ bekommen wir:

$$K^{ext} = \left(8|4\,7|3\,2|1\,4|2\,1|0\,3|2\,6|3\,5|3\right) \tag{4.31}$$

An diesem vollständigen Schlüssel können nun verschiedene Abhängigkeiten gezeigt werden. So gilt stets, dass der a/b-Wert kleiner als der jeweilige Spaltenschlüssel

ist. Dies ist leicht einzusehen, da einer Spalte nicht mehr überlange Spalten vorhergehen können, als überhaupt Spalten vorhanden sind. So wird die Schlüsselstelle, die eine Spalte auf die erste Position verschiebt, stets einen a/b-Wert von 0 aufweisen, da dieser keine anderen Spalten vorhergehen. Diese Einschränkung kann jedoch auch von der anderen Seite betrachtet werden: Der letzten Stelle müssen alle anderen überlangen Spalten vorausgehen. So wird die Schlüsselstelle die eine Verschiebung an die letzte Stelle bewirkt stets den Wert r oder $(r-1)$ aufweisen - abhängig davon ob es sich um eine normale oder überlange Spalte handelt.

Auch muss nach der Verteilung der überlangen Spalten beachtet werden, dass die $(m-1)$-langen Spalten verteilt werden. Angenommen der Würfel mit der Breite 8 würde $r = 7$ überlange Spalten besitzen, so existiert nur eine Spalte der Länge $m-1$. D.h. es ist undenkbar, dass mehr als zwei Schlüsselstellen den gleichen a/b-Wert aufweisen, da dies nur dann möglich ist, wenn mehr als eine Spalte diese Länge besitzt. Diese Einschränkung wird in folgender Mengennotation beachtet:

$$a/b \in [\ max(0, (k_s - 1) + (r - n))\ ,\ min((k_s - 1), r)\ ] \tag{4.32}$$

Dies soll nun exemplarisch für das Beispiel aus Abbildung 4.3 durchgeführt werden. Dieses betrachtet jedoch jede Schlüsselstelle alleine, ohne Rücksicht auf den restlichen Teil des Schlüssels. In Tabelle 4.1 lässt sich erkennen, dass je nach

Schlüsselwert	Mengenschreibweise	Werte für a/b
$k_s = 1$	$[\ max(0,-3)\ ,\ min(0,5)\ ]$	$\{0\}$
$k_s = 2$	$[\ max(0,-2)\ ,\ min(1,5)\ ]$	$\{0,1\}$
$k_s = 3$	$[\ max(0,-1)\ ,\ min(2,5)\ ]$	$\{0,1,2\}$
$k_s = 4$	$[\ max(0,0)\ ,\ min(3,5)\ ]$	$\{0,1,2,3\}$
$k_s = 5$	$[\ max(0,1)\ ,\ min(4,5)\ ]$	$\{1,2,3,4\}$
$k_s = 6$	$[\ max(0,2)\ ,\ min(5,5)\ ]$	$\{2,3,4,5\}$
$k_s = 7$	$[\ max(0,3)\ ,\ min(6,5)\ ]$	$\{3,4,5\}$
$k_s = 8$	$[\ max(0,4)\ ,\ min(7,5)\ ]$	$\{4,5\}$

Tabelle 4.1. Verhältnis zwischen Spaltenschlüssel und a/b-Wert

Würfelform nur bestimmte a/b-Werte bzgl. eines Schlüssels möglich sind. Ist der betroffene Spaltenschlüssel $k_i = 1$ wie in der ersten Zeile der Tabelle, so können dieser keine überlangen Spalten zuvorgehen. Ist der Spaltenschlüssel $k_i = 2$, so können dieser in diesem Fall maximal 0 oder 1 Spalte zuvorgehen. Betrachten wir die letzte Zeile $k_i = 8$, so ist an dieser Stelle klar, dass es bereits überlange Spalten gegeben haben muss und so schränkt sich die Anzahl auf 4 oder 5 ein. Wir betrachten erneut das Resultat:

$$K^{ext} = \big(8|4\ 7|3\ 2|1\ 4|2\ 1|0\ 3|2\ 6|3\ 5|3\big) \tag{4.33}$$

An der dritten Position des Schlüssels befindet sich die 2|1 und an der sechsten die 3|2. Diese zwei Spalten werden später offensichtlich nebeneinander liegen.

Daher ist nachvollziehbar, dass der a/b-Wert maximal um 1 ansteigen kann. Genauer gesagt kann dies hier sogar präzise bestimmt werden, da es sich bei der dritten Spalte (also der Schlüsselstelle 2|1) um eine überlange Spalte handelt. Folglich muss der a-Wert sogar genau um 1 ansteigen. Lösungen wie:

$$\begin{matrix}(0\,0\,2|1\,0\,0\,3|0\,0\,0)\\(0\,0\,2|1\,0\,0\,3|1\,0\,0)\\(0\,0\,2|1\,0\,0\,3|3\,0\,0)\end{matrix} \tag{4.34}$$

wären daher nicht möglich. Wir wollen nun erneut die Lösungen des Beispiels betrachten. Dafür hier erneut die Gleichungen aus 4.29:

$$\begin{matrix}(0\,6|1\,0\,0\,0\,0)\\\quad(1|0\,0\,0\,0\,0\,0)\\(0\,6|2\,0\,0\,0\,0)\\\quad(0\,1|0\,0\,0\,0\,0)\\(0\,6|3\,0\,0\,0\,0)\\\quad(0\,0\,1|0\,0\,0\,0)\\(0\,1|0\,0\,0\,0\,0)\\\quad(0\,0\,2|0\,0\,0\,0)\\(0\,2|0\,0\,0\,0\,0)\\\quad(0\,0\,0\,0\,0\,2|0)\end{matrix} \tag{4.35}$$

Auch ohne Kenntnis des Schlüssels können wir die ersten zwei Schlüssel ausschließen. Hierfür betrachten wir die Einschränkung der a/b-Werte aus dem vorherigen Abschnitt

$$a/b \in [\ max(0, (k_s - 1) + (r - n))\ ,\ min((k_s - 1), r)\ ] \tag{4.36}$$

mit $n = 6$, $r = 4$ und $k_s = 6$:

$$a/b \in [\ max(0, 3)\ ,\ min(5, 4)\ ] = \{3, 4\} \tag{4.37}$$

Damit ist 6|1 und 6|2 nicht möglich und kann aus dem Schlüsselraum entfernt werden. Das Ergebnis dieser Kryptanalyse lautet daher:

$$\begin{matrix}(0\,6|3\,0\,0\,0\,0)\\\quad(0\,0\,1|0\,0\,0\,0)\\(0\,1|0\,0\,0\,0\,0)\\\quad(0\,0\,2|0\,0\,0\,0)\\(0\,2|0\,0\,0\,0\,0)\\\quad(0\,0\,0\,0\,0\,2|0)\end{matrix} \tag{4.38}$$

Wir haben also nun aus einem Klar-/Chiffretext-Positionspaar 3 mögliche Schlüssel extrahiert, die jeweils eine Stelle in K1 und K2 bestimmen.

4.4 Schlüsselmerging und Konsistenzprüfungen

In den vorherigen Abschnitten wurde gezeigt, wie über ein Plaintext/Ciphertext Positionspaar eine Schlüsselmenge ausgerechnet und reduziert werden kann. In einem Beispiel wurde gezeigt wie hierbei der Schlüsselraum von möglichen Schlüsseln entsteht, die gleichberechtigt berücksichtigt werden müssen. Jeder Schlüssel in diesem Raum besitzt die gleiche Eintrittswahrscheinlichkeit. Liegen allerdings mehrere Paare vor, so müssen dementsprechend mehrere Schlüsselräume aufgestellt werden. Es gilt also nun durch Hinzufügen weitere Positionspaare:

- den restlichen Teil des Schlüssels zu finden und
- die Schlüsselräume weiter zu verkleinern um ein eindeutiges Ergebnis zu finden.

Zunächst müssen die Schlüsselräume miteinander verbunden werden. Während dieses Prozesses muss eine Schlüsselkonsistenzprüfung stattfinden: Die alten und neuen Schlüssel unterliegen gewissen Regeln die eingehalten werden müssen, ansonsten ist der Schlüssel ungültig und kann aussortiert werden. Falls keine Einschränkungen vorliegen, können die zwei Schlüsselräume zu einem neuen vereinigt werden.

4.4.1 Verbindung der Schlüsselräume

Im Beispiel aus Abschnitt 4.1 wurde das Positionspaar $14 \rightarrow 4$ analysiert und eine 3-elementige Lösungsmenge erzeugt. Wir bezeichnen diese Menge der möglichen Schlüssel als K^1. Analysieren wir nun in ähnlicher Weise ein anderes Paar, z.B. $19 \rightarrow 3$, so erhalten wir die Menge K^2. Da jedes Element dieser Mengen einen möglichen Schlüsselteil darstellt, muss jedes dieser mit jedem Element der anderen Menge verknüpft werden.

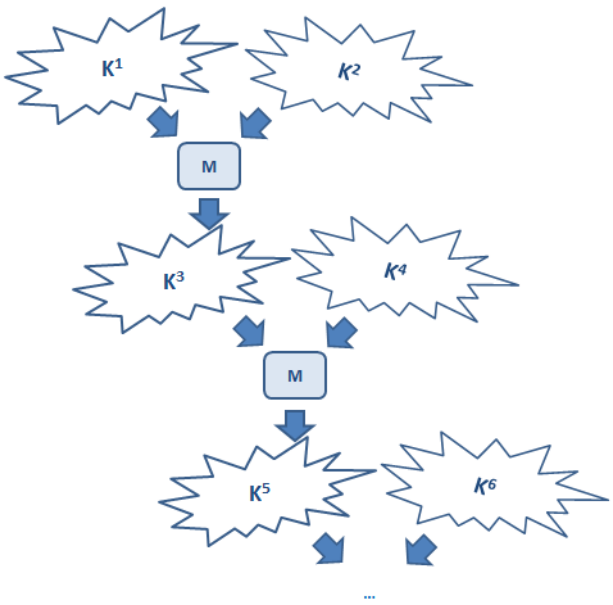


Abbildung 4.4: Grobablauf des Keymerging-Verfahrens

In Abbildung 4.4 ist dargestellt, wie dieses Keymerging eingesetzt werden kann. So wird zunächst von zwei Plain/Ciphertext Positionspaaren der mögliche Schlüsselraum ermittelt. Diese werden durch das Keymerging miteinander zu einer neuen Schlüsselmenge verbunden und dabei über Konsistenzprüfungen reduziert. Die entstandene Menge umfasst nun alle möglichen Schlüssel, die eine Verschiebung beider Positionspaare ermöglicht. Bei oberflächlicher Betrachtung gilt $K^3 = K^1 \times K^2$ also $|K^3| = |K^1| * |K^2|$. Jeder der Schlüssel in K^3 stellt eine mögliche Lösung dar. Um diese Lösungsmenge zu verkleinern, ist eine Verbesserung dieses Keymergings durch Konsistenzprüfungen notwendig.

4.4.2 Prüfung der Schlüsselkonsistenz

Um Schlüsselmengen auf Gültigkeit zu prüfen, müssen die Schlüssel während und nach dem Merging-Prozess auf Gültigkeit überprüft werden. Dafür müssen gewisse Einschränkungen beachtet werden, die nun Schritt für Schritt betrachtet werden sollen und einen sehr wichtigen Teil der Kryptanalyse darstellen. Nur mit ihnen ist es möglich, am Ende auf nur einen eindeutigen Schlüssel als Resultat zu kommen. Wir befinden uns nun in der Position, dass wir jeweils einen Schlüssel aus beiden Mengen betrachten. Sei also $K^1 = \left(k_1^1|a_1^1\ k_2^1|a_2^1\ k_3^1|a_3^1 \ldots k_n^1|a_n^1\right)$ und $K^2 = \left(k_1^2|a_1^2\ k_2^2|a_2^2\ k_3^2|a_3^2 \ldots k_n^2|a_n^2\right)$ aus K^2. Wir möchten diese zwei Repräsentanten der Mengen zu einem neuen Schlüssel verbinden.

Einmaligkeit

Da es sich bei den Schlüsseln um Permutationen handelt, darf jeder Spaltenschlüssel nur ein Mal innerhalb des Schlüssels vorkommen. Dies gilt sowohl für die Schlüssel aus den Schlüsselmengen, als auch für den neuen resultierenden Schlüssel

$$\forall l \in \mathbb{N} \forall i, j \in [1, n] :\ k_i^l = k_j^l \Leftrightarrow i = j \tag{4.39}$$

Beispiel:

$$\left(1\,2\,0\,3\,0\,5\,3\,0\right) \tag{4.40}$$

Da hier die 3 zweimal enthalten ist, kann dieser Schlüssel nicht gültig sein.

Ungleiche Schlüsselstellen

Sind zwei Spaltenschlüssel ungleich 0, so müssen diese identisch sein. Es ist nicht möglich zwei Schlüssel zu verbinden, die an der gleichen Position unterschiedliche Werte aufweist.

$$\forall i \in [1, n] :\ k_i^1 \neq 0 \wedge k_i^2 \neq 0 \Rightarrow k_i^1 = k_i^2 \tag{4.41}$$

Beispiel:

$$\begin{matrix}\left(0\,5\,0\,4\,0\,0\,2\,0\right)\\ \left(0\,5\,0\,4\,0\,0\,3\,0\right)\end{matrix} \tag{4.42}$$

Da hier an der vorletzten Stelle ungleiche Spaltenschlüssel vorliegen, können diese zwei Schlüssel nicht miteinander zu einem neuen Schlüssel kombiniert werden.

Identischer a/b-Wert

Ist ein Spaltenschlüssel (k_i) identisch, so muss der a/b-Wert ebenfalls identisch sein. Der a/b-Wert beschreibt die Anzahl der vorhergehenden überlangen Spalten und ist fest mit dem Spaltenschlüssel verbunden.

$$\forall i \in [1, n] : \; k_i^1 = k_i^2 \Rightarrow a_i^1 = a_1^2 \tag{4.43}$$

Beispiel:

$$\begin{matrix} \big(0\ 5|3\ 0\ 4|1\ 0\ 0\ 2|1\ 0\big) \\ \big(0\ 5|3\ 0\ 4|2\ 0\ 0\ 3|1\ 0\big) \end{matrix} \tag{4.44}$$

Da an der vierten Position die Spaltenschlüssel identisch sind, so müssen die dazugehörigen a/b-Werte auch identisch sein. Dies ist hier nicht der Fall und daher ist eine Kombination der Schlüssel nicht möglich.

Ansteigender a/b-Wert

Der a/b-Wert darf mit ansteigendem Spaltenschlüssel nicht sinken. Die Anzahl der vorhergehenden überlangen Spalten kann nur größer werden oder gleich bleiben, je weiter ich mich rechts im Würfel bewege.

$$\forall l \in \mathbb{N} \forall i, j \in [1, n] : \; k_i^l < k_j^l \Rightarrow a_i^l \leq a_j^l \tag{4.45}$$

Beispiel:

$$\big(0\ 5|2\ 4|3\ 0\ 0\ 0\ 0\ 0\big) \tag{4.46}$$

Die zweite Spalte wird zur fünften Spalte verschoben und besitzt zwei vorhergehende überlange Spalten. Die dritte Spalte wird zur vierten Spalte verschoben und gibt drei überlange Spalten an. Dies ist nicht möglich, da die Anzahl der vorhergehenden überlangen Spalten nicht sinken darf.

Verhältnis a/b-Wert zum Spaltenschlüssel

Ebenfalls müssen die Einschränkungen des a/b-Wertes in Bezug auf den Spaltenschlüssel eingehalten werden, die bereits in Abschnitt 4.3 behandelt wurden.

4.4.3 Vereinigung der Schlüssel

Hat die Schlüsselkonsistenzprüfung ein positives Resultat erbracht, so können die Schlüssel miteinander verbunden werden und bilden ein Element in der neuen Schlüsselmenge. Dieser Mergingvorgang i.V.m. den Konsistenzprüfungen kann fortgesetzt werden, bis alle Positionspaare beachtet wurden oder der Schlüssel vollständig errechnet wurde. Lassen sich zwei Schlüsselräume nicht miteinander vereinen, so entsteht die leere Menge und das Verfahren ist ebenfalls beendet. Falls der Schlüssel nur noch eine unbekannte Stelle aufweist, so kann diese vervollständigt werden.

4.5 Der Known-Plaintext Angriff

Bisher wurden nur Klartext/Ciphertext-Positionspaare betrachtet, also von welchem Index des Klartext, zu welchem Index des Chiffretextes verschoben wird. Diese Informationen können in einem Chosen-Plaintext Fall leicht erhoben werden, indem der Klartext binär durchnummeriert wird. Sofern jedoch nur ein festes Klar- und Chiffretextpaar bekannt ist, liegen diese Informationen u.U. nicht vor und müssen aus dem Text extrahiert werden.

Wir erinnern uns an das Eingangsbeispiel aus Abschnitt 1.1, indem der Klartext `HALLO DAS HIER IST EIN LANGER BEISPIELTEXT UM DAS VERFAHREN ZU ZEIGEN` (ohne Leerzeichen) in den Chiffretext `NRSGS ESAIE OZRAB INADI ILURT NDEHX USRHE VIEEP AEHEE GTLZF TLIAN MEL` verschlüsselt wurde. Die Information, an welche Position die erste Stelle des Klartextes verschoben wurde, liegt zunächst nicht vor. Dafür kommen sowohl 29, 34 also auch 43 in Frage. Es ist an dieser Stelle nicht möglich herauszufinden, welche dieser Stellen zutreffend ist. Falls, wie hier, mehrere Positionen in Frage kommen, müssen diese gleichbereichtigt behandelt und für jedes eine entsprechende Schlüsselmenge generiert werden. Nachdem dies abgeschlossen ist, muss dies ebenfalls für das zweite Zeichen durchgeführt werden und jede Schlüsselmenge für das erste Zeichen, muss mit jeder Schlüsselmenge für das zweite Zeichen verknüpft werden. Es ist offensichtlich, das bei so einem Vorgehen die Anzahl der möglichen Schlüssel deutlich wächst.

Um dies zu verhindern, wird zunächst eine Häufigkeitsanalyse auf den Chiffretext durchgeführt, um die am seltensten vorkommenden Zeichen zu finden. Tatsächlich kommen 7 Zeichen in diesem Beispiel nur 1-mal vor. Es handelt sich hierbei um das X, V, P, O, M, F und B. Damit wurden eindeutige Klartext/Chiffretext-Positionspaare gefunden und diese können sofort zu Schlüsselmengen verarbeitet werden:

$$\begin{aligned} 35 &\to 30 \\ 42 &\to 36 \\ 29 &\to 40 \\ 5 &\to 11 \\ 38 &\to 56 \\ 45 &\to 50 \\ 25 &\to 15 \end{aligned} \tag{4.47}$$

Mit diesen Informationen wäre es bzgl. dem Eingangsbeispiel möglich, die zwei Schlüssel beinahe vollständig zu rekonstruieren - es fehlen lediglich drei Stellen im ersten Schlüssel die mit dem Aufwand $3! = 6$ zu finden wären.

Liegt der vollständige Klar- und Chiffretext vor, so genügt eine einmalige Häufigkeitsanalyse, da sich die Anzahl der Buchstaben innerhalb von Transpositionschiffren nicht ändert. Ist allerdings nur ein geringer Teil des Klartextes bekannt, so ist es sinnvoll, nach Analyse des Chiffretextes auch eine Häufigkeitsanalyse auf den Klartext auszuführen um hier ebenfalls selten vorkommende Zeichen zu finden.

Mit zunehmender Textlänge bei konstantem Alphabet wird die Wahrscheinlichkeit für eindeutige Positionspaare wie oben gezeigt jedoch immer geringer. Aus diesem

Grund kann eine Behandlung wie oben geschildert nicht vermieden werden: alle Möglichkeiten müssen nacheinander untersucht werden.

Die Strategie dabei ist, möglichst selten vorkommende Zeichen zu bevorzugen, und sich so langsam von unten nach oben durch die Häufigkeitsanalyse zu arbeiten. In Abbildung 4.5 ist dieses Vorgehen aufgezeigt. So werden zunächst Schlüsselmengen mit den eindeutig vorkommenen Verschiebungen erzeugt, um diese im nächsten Schritt mit dem Schlüsselraum der zweifach vorkommenden Zeichen zu mergen; danach die dreifach vorkommenden Zeichen usw.

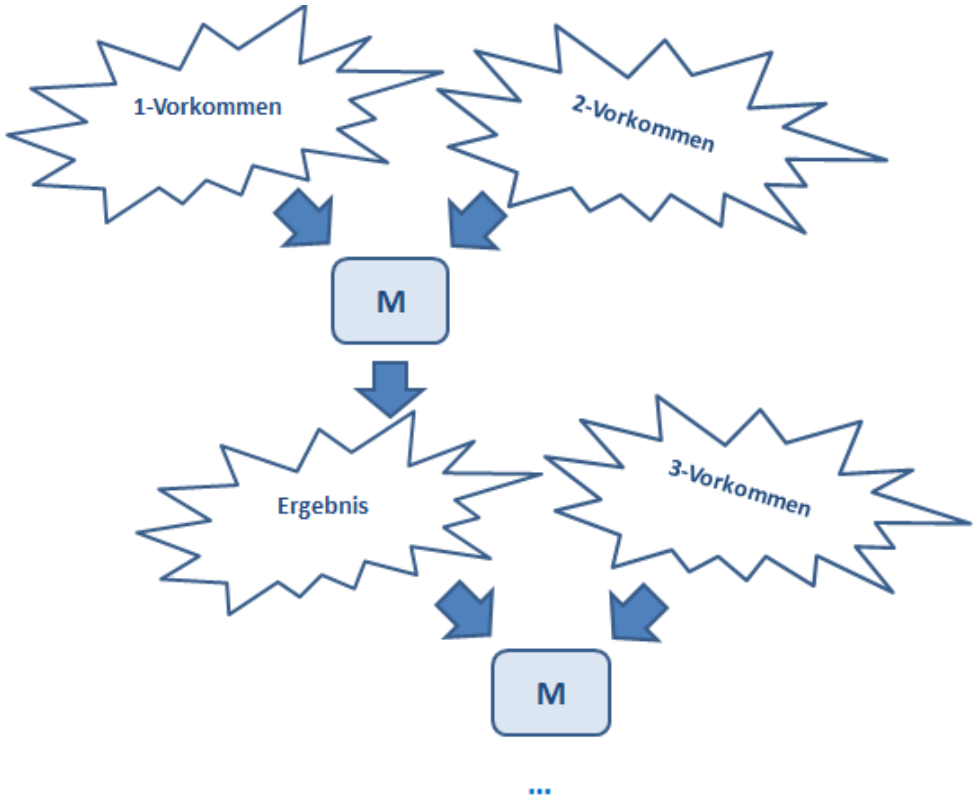


Abbildung 4.5: Mergingstrategie im Known-Plaintext Angriff: Verbindung von möglichst kleinen Schlüsselmengen

Es wurde ebenfalls ein Optimierungsweg gesucht, welcher die fehlenden Schlüsselstellen miteinbezieht. Angenommen innerhalb eines Schlüssels fehlt nur eine Stelle, so wäre es sinnvoll, gezielt nach Paaren zu suchen, welche eine Füllung dieser Stelle ermöglichen. Allerdings hat sich gezeigt, dass dieses Vorgehen nur gegen Ende einen kaum messbaren Mehrwert bringt und in den ersten Runden der Kryptanalyse die Laufzeit deutlich verlängert. Unter Umständen gäbe es jedoch an dieser Stelle noch weitere Ansätze die Paarbildung zu verbessern.

4.6 Zusammenfassung

In diesem Kapitel wurde der allgemeine Doppelwürfel betrachtet und analysiert. Dabei wurde mit Satz 4.6 eine Gleichung aufgestellt, welche die Verschlüsselung beschreibt. Die wesentliche Änderung zum vollständigen Würfel ist die Berücksichtigung der unterschiedlichen Spaltenlängen. Um diese zu kompensieren wurden die Variablen a und b eingeführt. Dieser a/b-Wert steht in einem direkten Verhältnis zum Spaltenschlüssel und erweitert diesen damit. Eine solche Erweiterung des Schlüssels ermöglicht es später so genannte falsche Schlüssel, also Schlüssel die unter falschen Annahmen erstellt worden sind, auszusortieren. Die Aussortierung findet innerhalb des Keymergings bzw. der Konsistenzprüfung statt. Diese Möglichkeit ist sehr wichtig, da in einem Known-Plaintext Fall eine große Anzahl von möglichen Schlüsseln in Frage kommt. Nur über diese Reduzierungsmöglichkeit ist eine Grundlage für einen solchen Angriff gegeben.

In diesem Kapitel nicht näher behandelt worden ist die Frage nach schwachen Schlüsseln für den allgemeinen Doppelwürfel. In Unterabschnitt 3.2.4 wurde dieses Thema bereits für den vollständigen Würfel behandelt. Durch den Verschiebeeffekt bei unvollständiger Spaltenfüllung, gelten die dort beschriebenen schwachen Schlüssel nicht für den allgemeinen Doppelwürfel. Jedoch sollte auch hier von der Verwendung eines solchen Schlüssels abgesehen werden, da Teile des Klartextes weiterhin im Chiffretext zu erkennen wären bzw. eine Kryptanalyse sich sehr einfach gestalten würde. Ebenfalls erwähnt sein sollte eine Information aus [Kul34], welche auf eine Wiederholung der indirekt definierten Permutationsfunktion φ aus Abschnitt 4.1 hinweist. Übersteigt die Länge des Klartextes das Produkt der Schlüssellängen $K1$ und $K2$, so ist die doppelte Spaltentranspositionschiffre als einfache Spaltentranspositionschiffre mit Schlüssellänge $|K| = |K1| * |K2|$ darstellbar.

5

Implementierung

In diesem Kapitel soll zunächst die Chiffre beispielhaft implementiert werden. Dies beinhaltet die Generierung eines Schlüssels aus einem Text, sowie die Permutation des Klartextes zum Chiffretext. Das Programm wird hier nur in Pseudocode beschrieben und dient allein der Vollständigkeit. Das Hauptaugenmerk dieses Kapitels liegt in der Umsetzung der Kryptanalyse. Hierbei soll eine Anwendung entwickelt werden, welche automatisiert einen Chosen-Plaintext oder Known-Plaintext Angriff durchführt. Der letzte Abschnitt beschäftigt sich mit einer grafischen Benutzeroberfläche, welche die Verwendung der Anwendung vereinfachen soll.

In dieser Arbeit werden nur die wesentlichen Stellen der Implementierung beschrieben. Die undokumentierten Stellen finden sich im Quelltext der Anwendung.

5.1 Der Chiffrierkern

5.1.1 Schlüsselgenerierung

Bereits in Abschnitt 2.3.1 wurde beschrieben, wie ein Schlüssel aus Textzeichen generiert werden kann. Dabei wurde die Position eines Zeichens im Alphabet berücksichtigt. Bei der Implementierung muss sich dabei jedoch mit einigen Problemen auseinander gesetzt werden. So ist die Frage, ob ein Unterschied zwischen Groß- und Kleinschreibung berücksichtigt werden muss. Ebenfalls ist zu überlegen, in wie weit Sonderzeichen als Teil des Schlüssels in Frage kommen. Sollte dies der Fall sein, muss ebenfalls eine Reihenfolge festgelegt werden. Aus der Literatur wurde allerdings der Eindruck ermittelt, dass Sonderzeichen - zumindest im Schlüssel - keine Verwendung finden. Aus diesen Gründen wurde an dieser Stelle ein eindeutiges Schlüsselalphabet festgelegt

$$\Sigma_K = \{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z\} \tag{5.1}$$

sowie eine Gewichtungsfunktion $\delta : \Sigma \to \mathbb{N}$:

$$\delta(A) = 1,\ \delta(B) = 2,\ \delta(C) = 3, \ldots\ \delta(Z) = 26 \tag{5.2}$$

Über diese Funktion kann die Umwandlung eines Schlüsseltextes in eine eindeutige Zahlenfolge ermöglicht werden. Bei Eingabe des Schlüsselwortes

$$K_T = \left(k_{T_1}\ k_{T_2}\ k_{T_3}\ \ldots\ k_{T_n} \right) \tag{5.3}$$

wird die geordnete Menge

$$K_Z = \left(\delta(k_{T_1})\ \delta(k_{T_2})\ \delta(k_{T_3})\ \ldots\ \delta(k_{T_n}) \right) \tag{5.4}$$

gebildet. In Tabelle 5.1 ist ein Beispiel für diesen Vorgang dargestellt.

K_T	H	A	L	L	O	W	E	L	T
K_Z	$\delta(H)$	$\delta(A)$	$\delta(L)$	$\delta(L)$	$\delta(O)$	$\delta(W)$	$\delta(E)$	$\delta(L)$	$\delta(T)$
K_Z	8	1	12	12	15	23	5	12	20

Tabelle 5.1. Beispiel zur Konvertierung von Text in Zahlenfolgen

Wir nehmen an, diese Folge wird in einem Feld *KZ* abgelegt. Im nächsten Schritt müssen wir daraus eine Permutation erzeugen, was bedeutet, dass es keine doppelten Einträge mehr gibt und die Lücken zwischen den Elementen geschlossen werden. Wir iterieren ausgehend von links über das Feld und zwar so oft, bis alle Zellen verarbeitet wurden. Dabei füllen wir das neue Feld K. Hierbei suchen wir stehts die Position des Minimums in KZ und verwenden diese Position in K um eine fortlaufende Zahl einzutragen.

```
1    zaehler := 1
2    for i:= 1 to length(KZ)        //Iteration über das Feld
3      posmin := getMinimumPos(KZ) //Suche nach dem Minimum
4      K[posmin] := zaehler         //Setzen des aktuellen Zählerstandes
5      zaehler := zaehler + 1       //Inkrementieren des Zählers
6      KZ[posmin] := -1             //Verwendete Felder markieren
7    next i
```

Die Funktion *getMinimumPos* gibt die Position des ersten Aufkommens des Minimums im Feld zurück, berücksichtigt dabei allerdings keine Einträge < 0. Die Arbeitsweise des Programms soll in Abbildung 5.1 verdeutlicht werden.

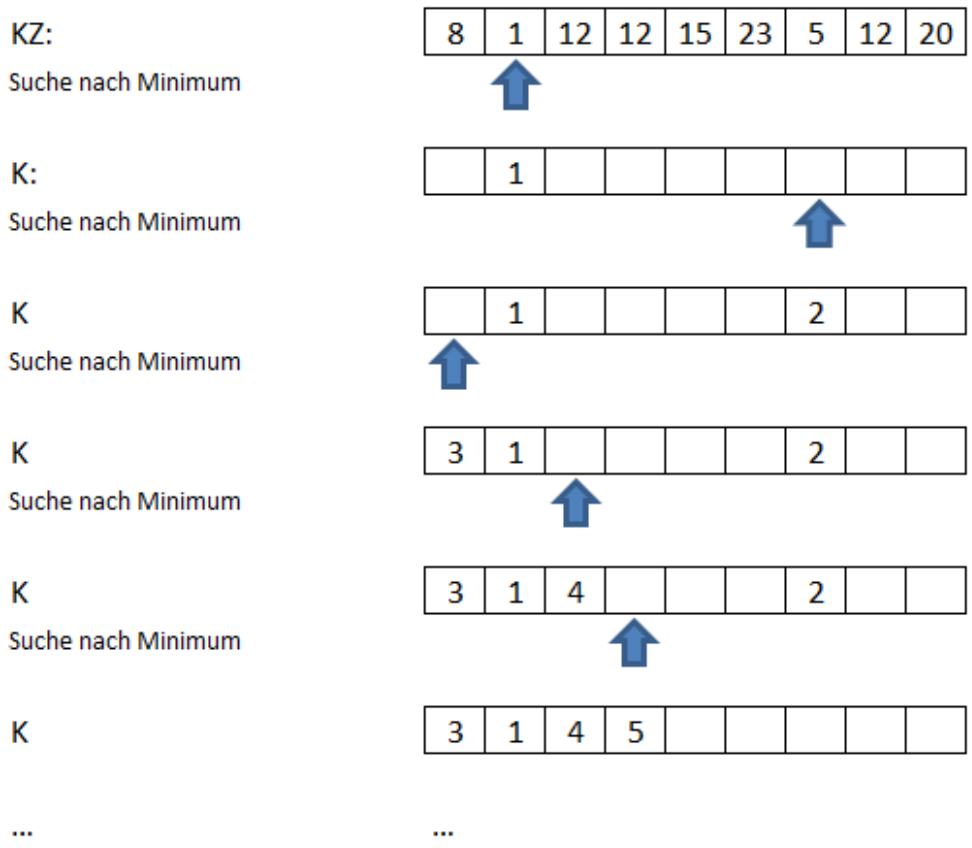


Abbildung 5.1: Schrittweiser Ablauf der Schlüsselerzeugungsroutine

Als Ergebnis erhalten wir: $\begin{pmatrix} 3 & 1 & 4 & 5 & 7 & 9 & 2 & 6 & 8 \end{pmatrix}$. Wir haben nun erfolgreich aus einem beliebigen Text eine eindeutige Permutation in einzeiliger Schreibweise generiert.

Anmerkungen zum Programm

Über die Verwendung einer doppeltverketteten Liste anstelle eines Feldes wäre eine in situ Implementierung möglich. Da es sich bei dem Schlüssel um Wörter oder kurze Phrasen handelt, wird sich die Schlüssellänge üblicherweise zwischen 5 und 30 bewegen. So steht der Mehraufwand einer solchen Lösung in keiner Relation. Aus diesem Grund wurde sich hier für diese naive Lösung mit der Zeitkomplexität $O(n^2)$ und Raumkomplexität $O(2n)$ entschieden.

5.1.2 Verschlüsselung

Nachdem der Schlüssel definiert wurde, soll nun anhand dessen ein beliebiger Klartext permutiert werden. Da es sich hierbei nur um eine Verschiebung des

Inhaltes handelt, gibt es in diesem Fall keine Einschränkung bzgl. dem Klartextalphabet. Wir betrachten folgende Elemente des Klartextes P

$$P = \left(p_1\ p_2\ p_3 \ldots p_s\right) \tag{5.5}$$

Wir verwenden den eben generierten Schlüssel aus dem Text „HALLOWELT“: $K = \left(3\,1\,4\,5\,7\,9\,2\,6\,8\right)$ und berücksichtigen zunächst nur die zweite Stelle, welche für die Verschiebung der zweiten Spalte an die erste Stelle verantwortlich ist. Die zweite Stelle des Klartextes verschiebt sich also an die erste Position des Chiffretextes:

$$C_1 = P_2 \tag{5.6}$$

Die zweite Stelle des Chiffretextes wird aus der $|K|$-sten Position des Klartextes entnommen. Daher gilt:

$$C_2 = P_{2+|K|} \tag{5.7}$$

also

$$C_2 = P_{2+9} = P_{11} \tag{5.8}$$

und dies analog für die dritte Stelle

$$C_3 = P_{2+2|K|} = P_{2+18} = P_{20} \tag{5.9}$$

solange, bis der Index von P die Länge des Klartextes übersteigt. Aus unseren Überlegungen von 2.3.3 wissen wir, dass die Anzahl der entstehenden Chiffretextzeichen pro Schlüsselposition von verschiedenen Faktoren abhängt. So endet dieser Vorgang bei m oder m-1, je nach Länge der Spalte bzw. Größe des Rechtecks. Sei hier m das Ende, so ist die Verarbeitung der ersten Schlüsselposition (zweite Position im Feld) mit C_m abgeschlossen. Und nun kann das Feld mit dem Inhalt 2 (siebte im Feld) verarbeitet werden.

$$\begin{aligned} c_{m+1} &= P_7 \\ c_{m+2} &= P_{7+|K|} = P_{7+9} = P_{16} \\ c_{m+3} &= P_{7+2|K|} = P_{7+18} = P_{25} \\ &\vdots \end{aligned} \tag{5.10}$$

Diese Erkenntnisse helfen bei der Implementierung der Verschlüsselung. Wie zu sehen ist, wird hierbei ein schneller indizierter Zugriff auf Felder notwendig. Aus diesem Grund eignet sich ein Array als Datenstruktur. Der Klartext liegt als Feld P vor und kann über den Index $1 \ldots |P|$ angesprochen werden. In vielen Programmiersprachen ist die 0 ebenfalls ein gültiger Index, was aber an dieser Stelle nicht berücksichtigt wird. Der Schlüssel liegt in Feld K vor, welches ebenfalls über den Index $1 \ldots |K|$ adressiert werden kann.

```
1    z:=1
2    for i:= 1 to length(K)
3      pos := getPos(K,i)          //Position von i in K
4      j:=0
5
6      //Schleife, bis Ende von P erreicht
7      while(pos + j+length(K) <= length(P))
8        C[z] := P[pos + j*length(K)] //Zusammensetzen von C
9        z := z + 1
10       j := j + 1
11     endwhile
12   next i
```

Auf diese Weise kann der neue Chiffretext aus dem Plaintext in Abhängigkeit des Schlüssels K generiert werden. Bemerkenswert an dieser Art der Implementierung ist, dass die Länge der Spalten (ob m oder $m-1$) nicht berücksichtigt werden muss. Dies ergibt sich automatisch aus der Überprüfung, ob die Länge des Feldes überschritten wurde. Zur Umsetzung der doppelten Spaltentransposition ist es notwendig, den Quelltext zwei mal auszuführen, wobei der Chiffretext des ersten Durchlaufs der Plaintext im zweiten ist.

5.1.3 Entschlüsselung

Die Entschlüsselung der Daten erfolgt analog. Dabei werden jeweils nur die verwendeten Felder vertauscht. Wo zuvor der Chiffretext über den Klartext zusammengesetzt worden ist, wird nun der Klartext durch den Chiffretext erzeugt.

```
1
2
3
4
5      ...
6       //Schleife, bis Ende von P erreicht
7      while(pos + j+length(K) <= length(P))
8        P[pos+j*length(K) := C[z] //Zusammensetzen von P
9        z := z + 1
10       j := j + 1
11     endwhile
12    ...
```

5.2 Implementierung der Kryptanalyse

In diesem Abschnitt soll die Kryptanalyse implementiert werden, die im vorherigen Kapitel durchgeführt wurde. Im Gegensatz zum Chiffrierkern wurde diese in der Programmiersprache Java implementiert. Die Kryptanalyse beinhaltet zunächst das Berechnen des Schlüsselraums und das Vereinen der Schlüsselräume bzw. die Reduktion. Es handelt sich hierbei um ein Proof of Concept (Machbarkeitsnachweis). Da die Verschlüsselung in der heutigen Zeit nicht mehr eingesetzt wird, spielt die Performanz eine untergeordnete Rolle. Viel mehr sollte hier von der Plattformunabhängigkeit profitiert werden, um das Verfahren (z.B. durch Integration eines Applets in einen Webauftritt) einer breiten Masse zur Verfügung zu stellen.

5.2.1 Berechnung des Schlüsselraums

In diesem Abschnitt soll Ausschnittweise gezeigt werden, wie der Schlüsselraum aus einem Chiffretext Ciphertext-Positionspaar berechnet werden kann. Bekannt sind hierzu die Länge des Textes und die Länge der verwendeten Schlüssel. Wir gehen davon aus, dass die Plaintextposition `i_alt` auf die Ciphertextposition `i_neu` verschoben wurde, z.B. die 14 auf die 4. In Abschnitt 4.1 wurde die Permutationsfunktion

$$(k2_{(k1_s-1)*(m_1-1)+a+z-t*|K2|} - 1) * (m_2 - 1) + b + (t + 1) \tag{5.11}$$

aufgestellt und anschließend vereinfacht. Diese Vereinfachung soll hier ebenfalls genutzt werden

$$v * (m_2 - 1) + b + (t + 1) \tag{5.12}$$

Zur Erinnerung: m_2 ist die Tiefe des Würfels und lässt sich aus Text- und Schlüssellänge berechnen. Der v-Wert ist ein Ergebnis aus der ersten Runde. Der b-Wert ist die Anzahl der überlangen Spalten. Beim t-Wert handelt es sich um die Zeile des Textes im Ausgangswürfel der zweiten Runde und bewegt sich daher zwischen 0 und $m_2 - 1$.

```
1
2           /*
3            *  Es gilt: v*(m2-1)+b+(t+1)
4            *  Erster Schritt: Iteration über mögliche v-Werte
5            *  Zweiter Scrhitt: Iteration über b-Werte
6            *                              &Ausschluss von unmöglichen
7            *  Dritter Schritt: Iteration über t-Werte
8            *  Vierter Schritt: Validierung des Ergebnisses
9            *  Hinzufügen in die Lösungsmenge
10           */
11          for (int v = 0; v < K2len; v++) {
12
13             /* Iteration über mögliche b-Werte */
14             for (int b = 0; b < v + 1 && b <= r2; b++)
15             {
16
17              /* Ausschluss von unmöglichen b-Werten */
18                  if (b >= v + (r2 - K2len))
19                  {
20
21                      /* Iteration über die möglichen t-Werte */
22                      for (int t = 0; t <= m2 - 1; t++) {
23
24                          /* Abschließende Validierung des Ergebnisses */
25                          if (v * (m2 - 1) + b + (t + 1) == i_neu) {
26
27                              /* Hinzufügen der Lösung in die Lösungsmenge für K2 */
28                              K2res.add(new int[] { v, b, t });
29                          }
30                      }
31                  }
32              }
33          }
```

Über dieses Schleifenkonstrukt werden gültige Lösungen für die zweite Runde der Verschlüsselung generiert. Es werden auch (in begrenztem Umfang) bereits Lösungen validiert, um fehlerhafte auszuschließen. Bei dem Objekt `K2res` handelt

es sich um eine einfach verkettete Liste welche die Lösungsmenge darstellt. Diese Datenstruktur ist geeignet, da hier eine variable Anzahl von Ergebnissen hinzugefügt wird. Ein gültiges Ergebnis ist ein Integer-Feld aus drei Elementen, die zur späteren Generierung des Schlüssels benötigt werden.

Damit ist die Kryptanalyse der zweiten Runde bereits abgeschlossen. Alle Elemente in dem Objekt `K2res` stellen zunächst gültige Lösungen dar, die erst später genauer geprüft werden. Enthält diese Liste nur ein Element, so ist uns bereits ein Spaltenschlüssel der zweiten Runde bekannt, da gilt:

$$v = k2_u - 1 \tag{5.13}$$

Wir kennen allerdings noch nicht die Position dieses Spaltenschlüssels, da u noch unbekannt ist. Dieses u muss nun berechnet werden.

Für jedes Element der Ergebnisliste `K2res`, also den potentiellen Kandidaten für den Schlüssel der zweiten Runde, werden nun die Gleichungen für die erste Runde aufgestellt. Wir suchen daher jetzt gültige Ergebnisse der ersten Runde, damit diese Gleichung erfüllt ist:

$$u = (k1_s - 1) * (m_1 - 1) + a + z - t * |K2| \tag{5.14}$$

Zur Erinnerung: Die Variable u ist die Position des Spaltenschlüssels des zweiten Schlüssels. Das $k1_s$ ist der Spaltenschlüssel der ersten Runde, wobei das s aus der Position des Plaintextzeichens und der Schlüssellänge berechnet werden kann. Das m_1 ist die Tiefe des Würfels der ersten Runde und ebenfalls bekannt. Der a-Wert repräsentiert die Anzahl der überlangen Spalten in der ersten Runde. Die Variable z beinhaltet die Zeile und kann ebenfalls berechnet werden. Das hier verwendete t ist ein Ergebnis, welches wir bereits im vorherigen Programmabschnitt eingeschränkt haben und das damit direkt eingesetzt werden kann.

```
1
2
3           /*
4            *  u = (k1_s-1)*(m1-1) + a + z - t*K2len
5            *  Schritt 1: Iteration über alle möglichen Lösungen für k1_s
6            *  Schritt 2: t-Ergebnis aus K2res einsetzen
7            *  Schritt 3: Anzahl der überlangen Spalten grob abschätzen
8            *  Schritt 4: Grobe Konsistenzprüfung
9            *  Schritt 5: Validierung von u
10           *  Schritt 6: Schlüsselmenge aufstellen
11           */
12          for (int k1_s = 1; k1_s <= K1len; k1_s++)
13          {
14
15              // t-Ergebnis aus K2res einsetzen
16              for (int[] k2r : K2res)
17              {
18                  int t = k2r[2];
19
20                  // Anzahl der überlangen Spalten grob abschätzen
21                  for (int a_s = 0; a_s <= r1; a_s++)
22                  {
23
24                      // Grobe Konsistenzprüfung
25                      if (a_s < k1_s && a_s >= (k1_s - 1) + (r1 - K1len))
26                      {
27                          int u = (k1_s - 1) * (m1 - 1) + a_s + z - (t * K2len);
28
29                          // Validierung von u
30                          if (u > 0 && u <= K2len) {
31
32                              // Schlüsselmenge aufstellen
33                              K1res.add(new int[] { s, k1_s, a_s, u });
34                          }
35
36                      }
37                  }
38              }
39          }
```

Im nächsten Schritt werden diese Lösungsmengen miteinander verbunden und zu einer eigenen Datenstruktur zusammengefasst. Die Abhängigkeit zwischen der Lösungsmenge `K2res` und `K1res` wird hierbei beachtet. Ebenfalls wird abschließend das Ergebnis erneut auf Erfüllung geprüft. Es gelangen also lediglich Ergebnisse in die Lösungsmenge, die auch tatsächlich diese Verschiebung bewirken können. Zu beachten ist an dieser Stelle, dass es zwischen dem Schlüssel K1 und K2 eine 1:n Beziehung gibt. So ändert sich, in Abhängigkeit des a-Wertes der ersten Runde, die Position des Spaltenschlüssels der zweiten Runde. Dieser Effekt konnte in Abschnitt 4.2 ebenfalls beobachtet werden.

5.2.2 Keymerging

Im vorherigen Abschnitt wurden Schlüsselmengen aufgestellt und anschließend in einer Datenstruktur zusammengefasst. In diesem Abschnitt gehen wir von der Situation aus, dass mehrere Plaintext Ciphertext-Paare zugrunde liegen. Für jedes dieser Paare wird eine eigene Schlüsselmenge bzw. ein eigener Schlüsselraum aufgebaut. Diese Schlüsselräume können gemäß Abschnitt 4.4 vereinigt und verkleinert werden, wofür ein Keymerging-Algorithmus benötigt wird. Aufgrund des Umfangs

kann der Algorithmus in Anhang A betrachtet werden und wird an dieser Stelle nur grob beschrieben.

Nachdem für z.B. zwei Paare jeweils ein Schlüsselraum aufgestellt wurde, wird diese merge-Funktion für jeden Schlüssel der einen Menge in Verbindung mit einem Schlüssel der anderen Menge aufgerufen. Danach werden erst die offensichtlichen Einschränkungen geprüft, wie z.B. das keine Schlüsselstellen doppelt vorkommen, da es sich um Permutationen handelt. Falls der Schlüsselinhalt nicht identisch ist, so wird anschließend geprüft, ob die a/b-Werte an den jeweiligen Stellen ebenfalls korrekt sind. Jeder Spaltenschlüssel besitzt einen eigenen festen a/b-Wert der sich in beiden Schlüsseln nicht unterscheiden darf. Anschließend wird geprüft, ob für steigende Spaltenschlüssel auch der a/b-Wert steigt. Nochmals der Hinweis: Der a/b-Wert kompensiert die Anzahl der vorhergehenden überlangen Spalten. Je weiter ich mich innerhalb des Würfels nach rechts begebe, so muss dieser Wert entweder gleich bleiben oder steigen - darf jedoch nicht sinken. Ebenfalls kann der a/b-Wert nicht größer werden, als die maximale Anzahl r, also dem Rest. In den detaillierten Konsistenzprüfungen wird erneut das Verhältnis zwischen a/b-Werten und dem Spaltenschlüssel betrachtet, wie sie in Abschnitt 4.3 beschrieben wurden. So gibt es in Bezug auf den Spaltenschüssel eine maximale Distanz zwischen a/b-Werten.

Falls alle diese Prüfungen positiv sind, können die zwei Schlüssel miteinander kombiniert und als möglicher Schlüssel innerhalb einer neuen Lösungsmenge betrachtet werden. Die Rückgabe der Methode ist entweder dieser neue Schlüssel - oder null, falls die Schlüssel nicht kombiniert werden konnten.

Nachdem jedes Element der einen Schlüsselmenge mit jedem Element der anderen Schlüsselmenge kombiniert wurde, enthält die neue Schlüsselmenge nur jene Ergebnisse, die beide Paarverschiebungen ermöglicht.

5.2.3 Paargenerierung aus Known-Plaintext

In Abschnitt 4.5 wurde erläutert, wie aus einem Klar- und Chiffretext Paare generiert werden können, die zum Aufbau einer Schlüsselmenge verwendet werden. Dabei wird die Häufigkeitsanalyse eingesetzt, um selten vorkommende Zeichen als erstes zu prüfen. In diesem Abschnitt soll dieser Algorithmus ausschnittweise skizziert werden.

Zum Anfang der Kryptanalyse gibt der Benutzer Klar- und Chiffretext ein. Diese werden dem so genannten *PairGenerator* übergeben. Es handelt sich hierbei um einen Generator, der auf Anfrage eine Liste von möglichen Paaren zurückgibt. Angenommen das Z ist im Klartext auf Position 48, 87, 98, und im Chiffretext auf Position 34, 77 und 83. An dieser Stelle würde der Paargenerator eine Liste der folgenden Werte zurückgeben:

- $48 \rightarrow 34$, $87 \rightarrow 77$, $98 \rightarrow 83$
- $48 \rightarrow 83$, $87 \rightarrow 34$, $98 \rightarrow 77$
- $48 \rightarrow 77$, $87 \rightarrow 83$, $98 \rightarrow 34$

Nur eine dieser drei Möglichkeiten ist korrekt, die anderen sind ungültig. Diese Generierung soll ein Algorithmus übernehmen.

Bevor wir uns der Generierung von solchen Paaren widmen, werden einige einmalige Vorberechnungen anstellt. So wird eine Häufigkeitsanalyse gestartet und damit selten vorkommende Zeichen gesucht. Dabei wird ein String *ciphFreqString* mit selten vorkommenden Zeichen befüllt. Für jedes dieser selten vorkommenden Zeichen wird die Position im Klartext gesucht.

```
1        int j=0;
2        /*
3         * ciphFreqString enthält eine sortierte
4         * Folge von selten vorkommende Zeichen
5         */
6        for (char c : ciphFreqString.toCharArray())
7        {
8           for (int i=0;i<klartext.length;i++)
9           {
10             /*
11              * Falls das Klartextzeichen ein
12              * seltenens Zeichen ist ...
13              */
14             if (klartext[i]==c)
15             {
16                /*
17                 * ... wird das Feld posList mit der
18                 * Position dieses Zeichens befüllt
19                 */
20                kList[j]=c;
21                posList[j]=delta+i;
22                j=j+1;
23             }
24          }
25       }
26       posList[j]=-1;
```

Das Array *posList* enhält also die Position von Zeichen im Klartext, die in Abhängigkeit ihrer Häufigkeit sortiert sind. Das bedeutet, dieses Feld würde im oben genannten Beispiel für das „Z“ jeweils einen Eintrag 48, 87 und 98 enthalten. Diese Vorberechnung muss nur ein Mal bei der Initiierung durchgeführt werden.

Im nächsten Schritt wird der Inhalt der Methode *getNextPairs()* genauer betrachtet. Als Ausgangsbasis besitzen wir das zuvor berechnete Feld *posList*. Die Idee ist, dass diese Methode immer den nächsten Eintrag in *posList* verarbeitet und die Ergebnisse zurückliefert. Verarbeitung bedeutet, dass die Position dieses selten vorkommenden Zeichens im Chiffretext gesucht wird.

Ein erster Aufruf der Methode *getNextPairs()* würde also die oben gezeigte Auflistung liefern, da der Algorithmus beim ersten Mal dreifach durchlaufen wird. Aus diesen Paarkombinationen kann eine Schlüsselmenge gebildet und entsprechend mit bereits vorhandenen gemergt werden.

```
1     public LinkedList<int[][]> getNextPairs()
2     {
3        /* Ergebnisliste*/
4        LinkedList<int[][]> result = new LinkedList<int[][]>();
5        int anz=1;
6
7        /*
8         * Beim ersten Durchlauf werden
9         * sofort mehrere Paare gebildet
10        */
11       if (pointer==0) anz=3;
12
13       for (int z=0;z<anz;z++)
14       {
15          /*
16           * posList enthält die Position eines Zeichens
17           * im Klartext. Diese Liste ist so sortiert, das
18           * selten vorkommende Zeichen am Anfang stehen.
19           */
20          if (pointer < kList.length && posList[pointer]!=-1)
21          {
22             char c=kList[pointer];
23             LinkedList<int[]> charResult = new LinkedList<int[]>();
24             int pos = posList[pointer];
25
26             /* Suche des Zeichens im Chiffretext  */
27             for (int j = 0; j < chiffretext.length; j++)
28             {
29                if (chiffretext[j] == c) {
30                   if (pos < klartext.length)
31                      /*
32                      * Die Paare gehen von 1-n,
33                      * das Array von 0-(n-1)
34                      */
35                      charResult.add(new int[] { (pos + 1), (j + 1) });
36                }
37             }
38             if (charResult.size() > 0) {
39                LinkedList<int[][]> newresult = combinate(result, charResult);
40                result = newresult;
41             }
42             pointer++;
43          }
44
45       }
46       return result;
47    }
```

5.2.4 Laufzeitabschätzungen

Um die Effizienz der Anwendung einzuschätzen, wurden verschiedene Laufzeitmessungen vorgenommen. Als Variablen gelten hierbei die Länge des Plaintextes, die Mächtigkeit des zugrundeliegenden Alphabets und die Länge des Schlüssels. Während Klartext und Schlüssellänge offensichtlich sind, muss die Begrifflichkeit des Alphabets erklärt werden. Das Alphabet spezifiziert die möglichen Zeichen im Plaintext. Besteht der Klartext nur aus Großbuchstaben ohne Sonderzeichen und Umlaute, so besitzt dieses die Mächtigkeit 26. Unterscheiden wir zwischen der Groß- und Kleinschreibung, erweitert sich dieses auf 52. Sind dagegen alle üblichen Zeichen eines deutschen oder englischen Textes möglich, so wird die Alphabetsgröße mit 78 beziffert.

|∑|: \ |K1|-|K2|:	|P| = 500				|P| = 1000				|P| = 1500			
	5-5	10-10	15-15	20-20	5-5	10-10	15-15	20-20	5-5	10-10	15-15	20-20
26 (A-Z)	1,732	4,6	18,7	94,45	2,2	8,1	15,2	55,4	0,57	5,1	13,9	48,1
52 (A-Z a-z)	0,2	1,09	2,9	11,4	0,2	0,9	4,4	11,1	0,7	2,8	7,6	7,7
78 (A-Z a-z 0-9 ...)	0,2	1,4	2,6	4,7	0,57	0,78	4,3	4,1	0,22	1,6	1,9	6,5

Angaben in Sekunden

Abbildung 5.2: Laufzeitmessung der Kryptanalysesoftware bei vollständigem Klar- und Chiffretextpaar

In Abbildung 5.2 sind drei verschiedene Chiffretextlängen mit jeweils 500, 1000 und 1500 Zeichen analysiert worden. Hierbei wurde zwischen den Schlüssellängen 5, 10, 15 und 20 unterschieden. Es handeln sich dabei um unabhängige Schlüssel K1 und K2. In den Messungen wurde ein Known-Plaintext-Angriff mit nur einem, jedoch vollständigen, Klar-/Chiffretextpaar durchgeführt, bei dem es sich um einen üblichen deutschen Text handelt. Die Ergebnisse werden erneut in Abbildung 5.3 grafisch dargestellt.

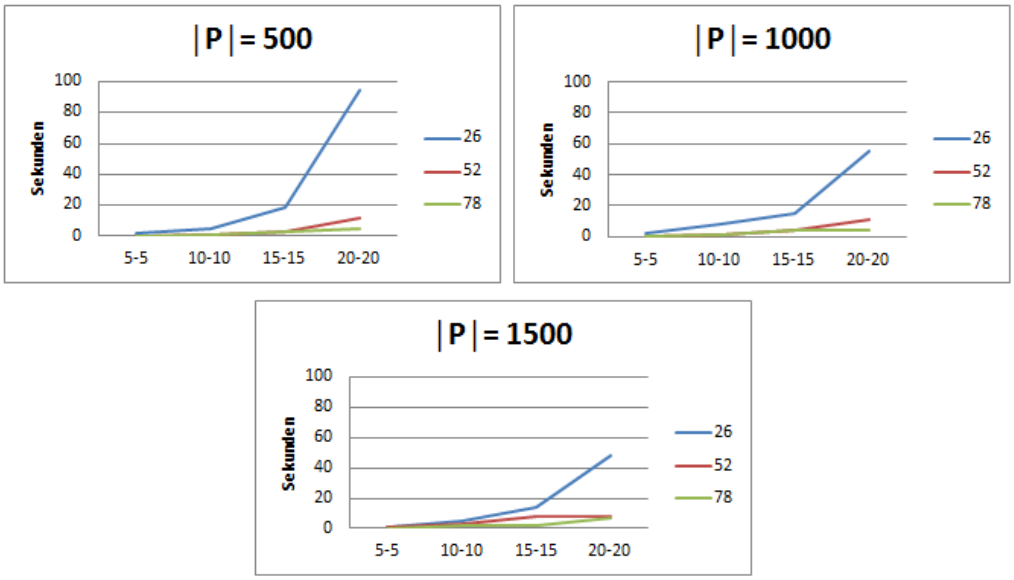


Abbildung 5.3: Visualisierung der Laufzeitmessung in einem Graphen

Es ist zu beobachten, dass die längste Berechnungszeit dann auftritt, wenn das Alphabet klein und die Textlänge kurz ist. Wir erinnern uns, dass innerhalb eines Known-Plaintext-Angriffs Klar-/Chiffretext-Positionspaare gebildet werden mussten. Je geringer die Mächtigkeit des Alphabets, desto unwahrscheinlicher sind eindeutige Informationen. Mit dieser Begründung kann der beobachtete Effekt leicht erklärt werden. Da in diesem Fall nicht zwischen Groß- und Kleinschreibung

differenziert wird, wird hier das „E“ am Wort- oder Satzanfang mit dem häufiger auftretenden „e“ in der Wortmitte gleichgesetzt. Diese Problematik scheint jedoch mit zunehmender Länge des Textes abzunehmen. Dies liegt daran, dass der Keymerging-Algorithmus schneller ungültige Schlüssel aussortieren kann, je mehr Informationen vorliegen. Eine zunehmende Textlänge verkürzt daher die Laufzeit deutlich.

Durch Vergrößerung des Alphabets ist die Herausfilterung eindeutiger Informationen aus dem Text wahrscheinlicher. So werden Sonderzeichen wie z.B. ein Semikolon innerhalb eines normalen deutschen Textes nur sehr selten verwenden. Tritt dieses im Klartext nur 1-mal auf, so erhalten wir sofort ein Positionspaar. Aus den Abbildungen ist sehr deutlich der Einfluss der Alphabetsgröße zu erkennen.

Zwar ist durchaus einen Einfluss der Schlüssellänge festzustellen, jedoch muss berücksichtigt werden, dass eine Schlüssellänge > 20 als sehr unwahrscheinlich anzusehen ist, da es sich bei dem Schlüssel um ein Wort oder eine kurze Phrase handelt.

	|P| = 500				|P| = 1000				|P| = 1500			
|K1| - |K2|:	5-5	10-10	15-15	20-20	5-5	10-10	15-15	20-20	5-5	10-10	15-15	20-20
|∑|:												
26 (A-Z)	2,07	11,82	39,7	483	1,83	15,38	298,6	1050	1,7	26,1	24,7	3317
52 (A-Z a-z)	0,31	3,4	8,2	74,2	0,47	2,1	40,04	60,8	0,83	6,5	28,75	387
78 (A-Z a-z 0-9 ...)	1,7	2,42	38,69	105	0,38	1,16	36,2	29,2	0,38	4,89	36,58	170,5

Angaben in Sekunden

Abbildung 5.4: Laufzeitmessung der Kryptanalysesoftware bei der Hälfte des Klartextes

Die Abbildung 5.4 zeigt die Laufzeit, wenn nur die Hälfte des Klartextes gegeben ist. Auch hier ist sehr deutlich die Mächtigkeit des Alphabetes als Einflussfaktor zu sehen. Allerdings liegen auch hier die Laufzeiten in einem Rahmen, der durchaus zu vertreten ist. Es wird gezeigt, dass sogar nur ein Teil des Klartextes genügt um eine Entschlüsselung durchzuführen. Weitere Versuche ergaben, dass bei ca. einem Fünftel des Klartext, die erfolgreiche Rekonstrution des Schlüssels in weniger als einer Stunde durchführbar ist. Dies auch dann, wenn nur ein kurzer Text und eine Alphabetsgröße von 26 Zeichen vorliegt.

In diesem Abschnitt wurde dargestellt, wie sich die Laufzeit in Abhängigkeit der Variablen Textlänge, Alphabetgröße und Schlüssellänge verhält. Bei den verwendeten Texten handelte es sich um zufällig gewählte Repräsentanten aus diesen Äquivalenzklassen. So kann es bei der Verwendung von anderen Klartexten durchaus zu Abweichungen kommen.

5.3 Frontend

Um die Verwendung der Software zu erleichtern, wurde eine grafische Oberfläche unter Verwendung von Swing/AWT aufgebaut. Diese ermöglicht die Eingabe von Klar- und Chiffretext sowie die Auswahl der Schlüsselbreiten. Ziel ist es ebenfalls, die Anwendung als Applet bereitzustellen und so innerhalb des Browserfensters ausführbar zu machen.

5.3.1 Oberflächendesign

Zur Komposition der notwendigen Schaltflächen wurde der JFormDesigner[1] (Abbildung 5.5) verwendet. Damit können die notwendigen Bedienelemente auf der Oberfläche platziert werden. Der Benutzer muss allerdings schon Vorkenntnisse aus dem Bereich der GUI-Programmierung mitbringen, um die notwendigen Einstellungen tätigen zu können. Am Ende kann der Java-Quelltext extrahiert und in das eigene Programm integriert werden.

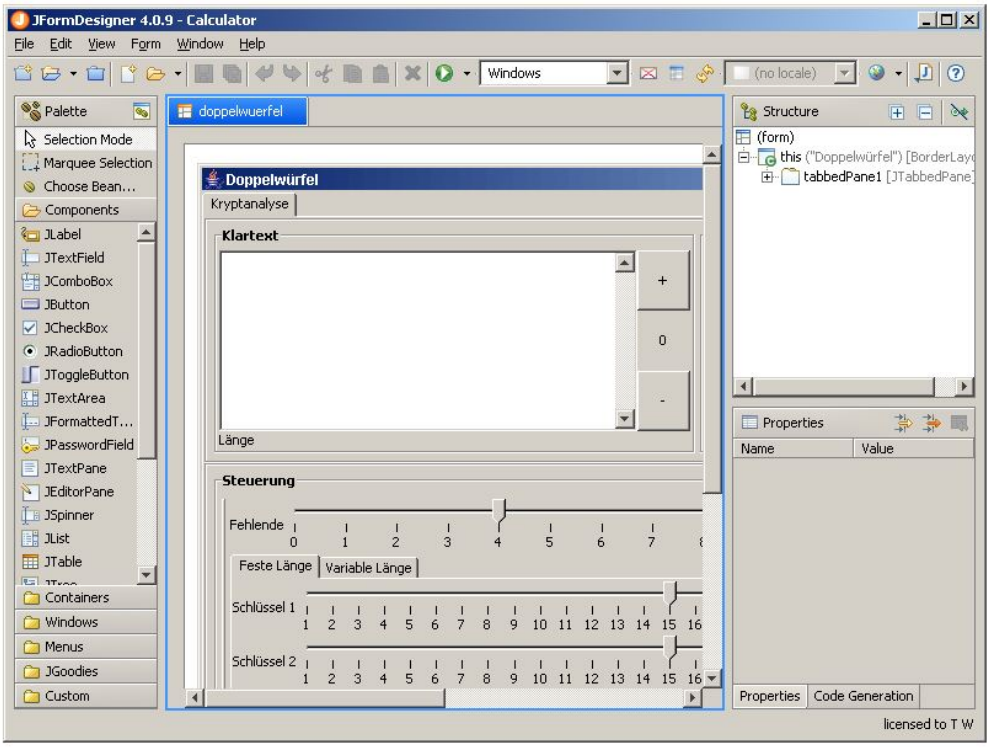


Abbildung 5.5: Gestaltung der Oberfläche im JFormDesigner

Bei der Konzeptionierung wurde auf die einfache Bedienbarkeit geachtet. Die Eingabe des Klar- und Chiffretextes erfolgt über normale Textfelder, die gleichzeitig die Länge des Textes angeben. Die Schlüssellängen können über Schieberegler eingestellt werden. Die Ergebnisse werden in Text- und Tabellenform präsentiert.

[1] http://www.formdev.com/

Um die Laufzeit zu verkürzen, kann über den „Missing“-Regler die Anzahl der maximal fehlenden Schlüsselstellen erhöht werden. Das Programm ermöglicht es, sowohl die Schlüssellängen fest anzugeben, als auch über Brute Force alle gängigen Schlüssellängen zu testen. Diese zwei Optionen spiegeln sich in Reiterform in der Mitte des Fensters wider.

5.3.2 Das MVC-Modell

Im Aufbau der grafischen Oberfläche wurde das MVC-Modell[2] angewendet. Dabei wird üblicherweise zwischen den Elementen für die Eingabe, zur Ausgabe und einem Modell zur Organisation und Zustandsverwaltung unterschieden.

Als Modell dient in dieser Anwendung die Klasse *DoppelwuerfelModel*, welche als Schnittstelle zwischen der grafischen Oberfläche und dem Kern der Kryptanalyse dient. Die grafische Oberfläche wird in der Klasse *Doppelwuerfel* erzeugt. Mit Betätigung der Buttons auf der GUI wird über den *DoppelwuerfelHandler* ein Signal an das Modell weitergegeben. Übliche Interfaces hierfür sind *ChangeListener* und *ActionListener*. Der Handler verarbeitet alle Eingaben auf der Oberfläche und gibt diese, sofern notwendig, an das Modell weiter.

Nach Verarbeitung im Modell wird ggf. ein Zustandswechsel durchgeführt. Sind alle notwendigen Daten vorhanden, wird innerhalb des Modells der Kryptanalyseprozess über die Klasse *DoppelwuerfelThread* gestartet. In diesem Thread wird auf ein Objekt der Klasse *Calculator* aus dem Package *core* zugegriffen. Dieses ist u.A. für die Generierung von Paaren zuständig wie in Unterabschnitt 5.2.3 erklärt wurde. Aus der Generierung von Paaren erfolgt der Aufbau der Schlüsselmenge gemäß Unterabschnitt 5.2.1.

Das Modell erhält stetig Informationen bzgl. des aktuellen Status zurück und leitet diese an die grafische Ausgabeelemente weiter. Dafür sind View-Elemente wie z.B. das *StatusListModel* oder das *ResultTableModel* zuständig. Sie zeigen aktuelle Rundenergebnisse und das Gesamtergebnis am Ende der Ausführung.

[2] MVC steht für Model-View-Controller

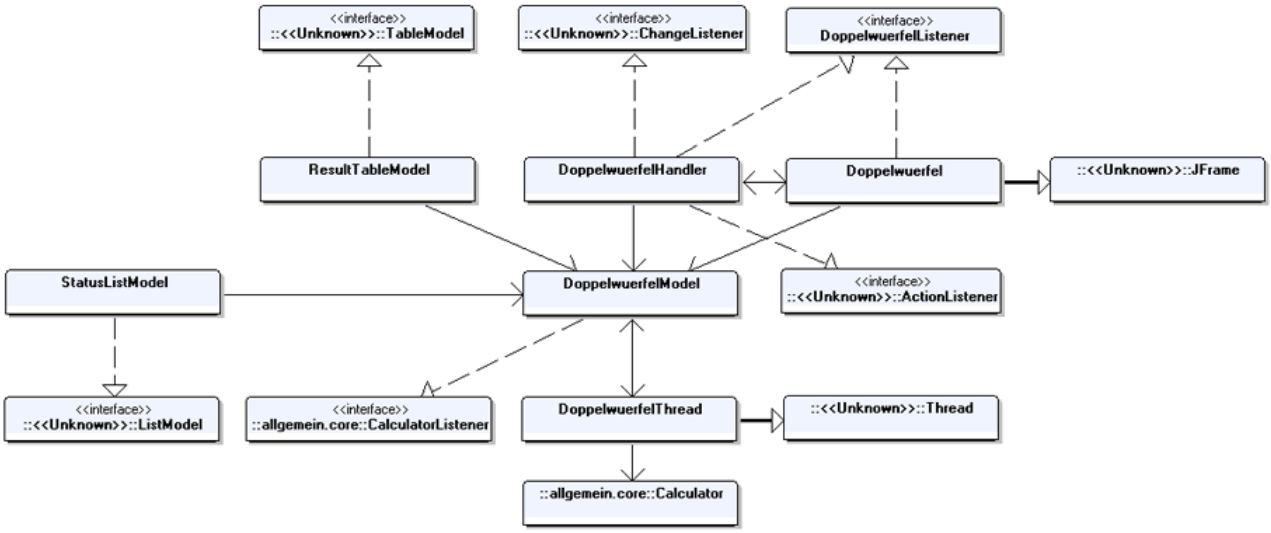


Abbildung 5.6: Analyse-Klassendiagramm der grafischen Oberfläche

6

Zusammenfassung

In Abschnitt 1.4 wurden für diese Arbeit folgende Ziele festgelegt:

- Eine Übersicht der Chiffre bieten und zu anderen Verschlüsselungen abgrenzen.
- Den mathematischen Hintergrund von Permutationschiffren durchleuchten.
- Suche nach Ansätzen zur Kryptanalyse der Verschlüsselung.
- Spezifizierung eines Chosen-Plaintext und Known-Plaintext Angriffs.
- Implementierung der Kryptanalyse in der Programmiersprache Java.
- Bereitstellung einer grafischen Oberfläche als Java-Applet.

Diese Arbeit hat es sich zunächst zum Ziel gesetzt, eine Übersicht bzgl. des Doppelwürfels zu liefern. Es wurden die besonderen Eigenschaften der Verschlüsselung hervorgehoben, so dass eine Abgrenzung zu anderen Verfahren möglich ist. Bei der Sichtung von Quellen und verwandten Arbeiten wurde festgestellt, dass diese keine klaren Wege aufzeigen, eine Kryptanalyse der Chiffre vorzunehmen. Insbesondere zeigte sich, das bisher noch kein Verfahren für einen automatisierten Angriff entwickelt wurde.

Um diese Lücke zu schließen, wurde sich zunächst mit dem mathematischen Hintergrund der Verschlüsselung beschäftigt. So wurde die Verschiebung von Klar-Chiffretextpositionen nur über eine einzige Gleichung beschrieben. Mit Einsetzen der jeweilen Variablen, z.B. K1 und K2, wurde eine Permutationsfunktion definiert, welche zur Durchführung der Verschlüsselung in der Lage ist. So konnte für eine beliebige Position im Klartext die entsprechende Position im Chiffretext errechnet werden.

Zur Kryptanalyse wurde dieses Vorgehen umgedreht. Die Verschiebungen zwischen Klar- und Chiffretext seien also als bekannt vorausgesetzt, so sollen darüber Rückschlüsse auf K1 und K2 getroffen werden. Diese Informationen sind jedoch nicht stets eindeutig und so müssen zunächst Lösungsmengen aufgebaut werden, in denen jedes Element als möglicher Schlüssel anzusehen ist. Es wurde ein Verfahren konzipiert, welches verschiedene Lösungsmengen miteinander kombiniert und dabei versucht den Schlüsselraum zu reduzieren.

Das skizzierte Verfahren beschreibt jedoch nur einen Chosen-Plaintext Angriff. Viel wahrscheinlicher ist allerdings, das lediglich ein Klar-/Chiffretextpaar oder sogar nur ein Teil des Klartextes vorliegt. Aus diesem Grund musste eine Strategie erarbeitet werden, welche einen Known-Plaintext Angriff ermöglicht. Hierbei wurde

sich auf ein recht altes Verfahren gestützt, um eindeutige Verschiebeinformationen aus dem Textpaar zu erlangen: die Häufigkeitsanalyse. Tritt ein Zeichen nur 1-mal innerhalb des Klar- und Chiffretextes auf, so kann daraus direkt ein Positionspaar abgeleitet werden. Diese Information ermöglicht es, weitere Schlüsselinformationen dem Verfahren hinzuzufügen und ggf. die bestehende Schlüsselmenge zu verkleinern. Die Reduzierung von Schlüsselmengen ist dadurch verbessert worden, indem dem Schlüssel eine weitere Komponente hinzugefügt worden: der a/b-Wert. Über diesen a/b-Wert ist es möglich, ungültige Schlüssel schneller zu erkennen und auszusortieren. Dies wäre vorher nicht möglich gewesen und so wäre der Schlüsselraum stetig gewachsen bzw. nicht länger beherrschbar gewesen. Die Eigenschaft der unterschiedlichen Spaltenlängen war es im Grunde, welche eine Analyse so effizient ermöglicht hat.

Nachdem diese theoretischen Grundlagen erarbeitet waren, wurde das Verfahren implementiert. Die Programmiersprache Java bot sich aufgrund ihrer Plattformunabhängigkeit an. Um den Algorithmus komfortabel nutzen zu können, wurde eine grafische Benutzeroberfläche implementiert. Abschließend wurden Laufzeitmessungen vorgenommen, um die Effizenz der Analyse zu zeigen.

6.1 Ausblick

Nicht behandelt wurde in dieser Arbeit die Frage, in welcher Weise die Sicherheit des Doppelwürfels verstärkt werden kann. So wären wohlmöglich die mehrfache Anwendung der Verschlüsselung bzw. die Erweiterung um weitere Runden denkbar. Hier wäre jdoch die Gefahr, dass eine nachfolgende Runde eine vorherige aufhebt. Ob die Kaskadierung der Chiffre, wie z.B. beim 3DES, einen weiteren Zuwachs an Sicherheit bietet, wäre dabei zu prüfen.

Ebenfalls wäre die Neuimplementierung in einer hardwarenahen Programmiersprache möglich. Zwar wurde in der Implementierung die Verwendung von mehreren Prozessorkernen berücksichtigt, doch wäre auch eine Erweiterung zum Betrieb als verteilte Anwendung denkbar.

Auch nicht näher betrachtet wurden Optimierungsmethoden, welche die verwendete Sprache innerhalb des Klartextes berücksichtigen. Da manche Zeichenkombinationen innerhalb eines Textes mit bekannter Sprache ausgeschlossen werden können, wäre so ggf. eine weitere Reduzierung des Schlüsselraums möglich.

Offen geblieben ist ebenfalls die Frage nach einem Ciphertext-only Angriff. Bisherige Untersuchungen bzgl. einer Erweiterung des Verfahrens zeigten, dass der Schlüsselraum (Menge möglicher Schlüssel) bei zu kurzen gegebenen Klartexten immernoch zu groß ist. Die Idee war, über eine Häufigkeitsanalyse seltene Buchstaben im Chiffretext zu finden und diese mit einem Wörterbuch abzugleichen. Zeigte sich beispielsweise, dass im Text weder „j“ noch „z“, dafür jedoch genau ein „q“ vorkommt, so können über ein Wörterbuch alle möglichen Wörter gesucht werden die diesem Kriterium entsprechen. Es zeigten sich ca. 2000 Wörter der englischen Sprache die solch eine Eigenschaft aufwiesen. Diese hatten eine durchschnittliche Länge von 8 Zeichen. Da pro Zeichen jedoch maximal eine Schlüsselstelle wiederhergestellt

werden kann, wird eine Suche erst ab ca. 25 Zeichen sinnvoll. Das bedeutet, es müssten mindestens drei Wörter gefunden werden, die im Klartext vorgekommen sind. Da jedes wahrscheinliche Wort mit einem anderen kombiniert werden muss, kommen wir an dieser Stelle auf 200^3 Verknüpfungen. Im nächsten Schritt muss die Position jedes dieser Wörter geraten werden. Bei einem ca. 600 stelligen Text entspricht dies dem Aufwand von etwa 600^3. Anschließend muss die Schlüssellänge gesucht werden. Sofern diese im Bereich von 20-25 liegt, wäre dies ein Aufwand von 25^2. Wir kommen also auf 1080000000000000000 Prüfungen. Es ist nicht zu erwarten, dass eine einzige Prüfung eines so kurzen Klartextes weniger als 24 Stunden benötigt. An dieser Stelle wird offensichtlich, dass sich der Algorithmus nicht für eine solche Analyse eignet. Sobald jedoch Teile des Klartextes, oder sogar Verschiebungen (also Positionspaare) bekannt werden, ist dieses Verfahren die bisher erfolgreichste automatisierte Kryptanalyse der doppelten Spaltentranspositionschiffre.

A

Anhang A

Der Keymerging-Algorithmus aus Unterabschnitt 5.2.2.

```
1
2     private static Key merge(Key k1, Key k2, int r)
3     {
4        boolean merge=true;
5        Key k_new = new Key();
6        k_new.k = new int[k1.k.length][2];
7
8        for (int i_1=1;i_1<k1.k.length&&merge;i_1++)
9        {
10          for (int i_2=1;i_2<k2.k.length&&merge;i_2++)
11          {
12             if (k1.k[i_1][0] != 0 && k2.k[i_2][0] != 0)
13             {
14                if (k1.k[i_1][0] == k2.k[i_2][0]) //falls key-inhalt identisch
15                {
16                   /*
17                    * falls schlüssel identisch, müssen a/b Werte
18                    * stimmen und die Position gleich sein
19                   */
20                   if (k1.k[i_1][1] != k2.k[i_2][1] || (i_1 != i_2))
21                      merge=false;
22                } else {
23                   // Falls key-inhalt nicht identisch
24                   if (i_1==i_2)
25                      merge=false;
26
27                   /*
28                    * Falls k_i>k_j gilt, so muss auch a_i>=a_j gelten
29                    */
30                   if (k1.k[i_1][0] > k2.k[i_2][0] && k1.k[i_1][1]<k2.k[i_2][1])
31                      merge = false;
32                   if (k1.k[i_1][0] < k2.k[i_2][0] && k1.k[i_1][1]>k2.k[i_2][1])
33                      merge = false;
34
35                   /*
36                    * Falls die Spalte eine Überlänge besitzt (i<r),
37                    * so kann der a/b-Wert nicht r sein
38                    */
39                   if (i_1<=r && k1.k[i_1][1]==r)
40                      merge=false;
41                   if (i_2<=r && k2.k[i_2][1]==r)
42                      merge=false;
43                }
44             }
45          }
```

```
1           if (merge)
2           {
3              if (k1.k[i_1][0]!=0)
4                 k_new.k[i_1]=k1.k[i_1];
5              if (k2.k[i_1][0]!=0)
6                 k_new.k[i_1]=k2.k[i_1];
7           }
8        }
9
10       //Weitere Konsistenzprüfungen
11       for (int i=1;i<k_new.k.length;i++)
12         {
13            for (int j=1;j<k_new.k.length;j++)
14            {
15
16               if (k_new.k[i][0]!=0 && k_new.k[i][0]!=0)
17               {
18               /*
19                * Wenn der Schlüsselwert k in einer überlangen Spalte sitzt
20                * dann gilt:
21                * Schlüsselwert +1 -> a+1
22                * Schlüsselwert +2 -> a+1,2
23                * Schlüsselwert +3 -> a+1,2,3
24                */
25                  if (k_new.k[i][0]<k_new.k[j][0] )
26                  {
27                     //Betrachten des Schlüsselwertabstands
28                     int delta_k = k_new.k[j][0] - k_new.k[i][0];
29
30                     //Betrachten des a/b-Abstands
31                     int delta_a = k_new.k[j][1] - k_new.k[i][1];
32
33                     // der Abstand von a kann nicht größer sein
34                     if (delta_a>delta_k)
35                        merge=false;
36
37                     // falls es sich um eine überlange spalte gehandelt hat
38                     if (i<=r)
39                     {
40                        if (delta_a==0) //kann der Abstand nicht 0 sein
41                           merge=false;
42                     }
43
44
45                  }
46
47               }
48            }
49       }
50
51       if (merge)
52          return k_new;
53
54       return null;
55    }
```

Literatur

Bar95. Barker, Wayne G.: *Cryptanalysis of the Double Transposition Cipher*. Aegean Park Press, 1995.

Bau07. Bauer, Friedrich L.: *Decrypted Secrets*. Springer, 2007.

Bec09. Beckman, Bengt: *Arne Beurling und Hitlers Geheimschreiber*. Springer, 2009.

Beu09. Beutelspacher, Albrecht: *Kryptologie*. Vieweg+Teubner, 2009.

Fri41. Friedman, William F.: *Military Cryptanalysis*. War Department, 1941.

JM07. Jiri Matousek, Jaroslav Nesetril: *Diskrete Mathematik*. Springer, 2007.

Kip06. Kippenhahn, Rudolf: *Verschlüsselte Botschaften*. Nikol Verlagsgesellschaft, 2006.

Kri13. Kriegsministerium: *Anleitung zur Geheimschrift innerhalb des Heeres*, 1913.

Kul34. Kullback, Solomon: *General Solution for the Double Transposition Cipher*. Aegean Park Press, 1934.

Sal05. Salomon, David: *Coding for Data and Computer Communications*. Springer, 2005.

Sch08. Schmeh, Klaus: *Codeknacker gegen Codemacher: Die faszinierende Geschichte der Verschlüsselung* . W3l, 2008.

Sin66. Sinkov, Abrahab: *Elementary Cryptanalysis*. The Mathematical Association, 1966.

Sin01. Singh, Simon: *Geheime Botschaften*. dtv, 2001.

Sin09. Singh, Simon: *Codes - Die Kunst der Verschlüsselung*. dtv, 2009.

Sta07. Stamp, Mark: *Applied Cryptanalysis*. Wiley, 2007.

Tes06. Teschl, Gerald: *Mathematik für Informatiker*. Springer, 2006.