

**Kryptoanalyse der Enigma-Maschine: Eine Untersuchung  
des Cyclometers und der Turing-Bombe**

**Kryptoanalyse der Enigma-Maschine**

Emanuel Schäffer

18. Oktober 2024

RWU–University of Applied Sciences

Enigma-Nachrichten können dechiffriert werden, wenn die Rotorauswahl, die Rotor-, Ringstellungen und die Steckereinstellungen einzeln wiederhergestellt werden. Die Wiederherstellung der Nachrichtenschlüsseleinstellung ist empfindlich genug, um die korrekte Rotorreihenfolge zu unterscheiden.

# 1 Enigma

## 1.1 Einführung

Die Enigma ist eine Rotor-Chiffriermaschine, die hauptsächlich im Zweiten Weltkrieg Einsatz kam. Die von der Wehrmacht modifizierte Enigma verfügte zunächst über drei Walzen und zusätzlich einen Reflektor. Jede dieser Walzen führte eine monoalphabetische Substitution durch. Die rechte Walze wurde bei jedem Tastendruck um eine Position weitergerückt. Hat diese Walze eine komplette Rotation vollzogen, rückte die Walze links neben ihr um eine Position weiter<sup>1</sup>. Neuerungen waren das Steckerbrett (siehe Abb. 1.1) und gegen Ende des Krieges eine zusätzliche, vierte Walze. Ich werde mich, wenn nicht ausdrücklich erwähnt, auf die Enigma M3 mit drei Walzen beschränken.

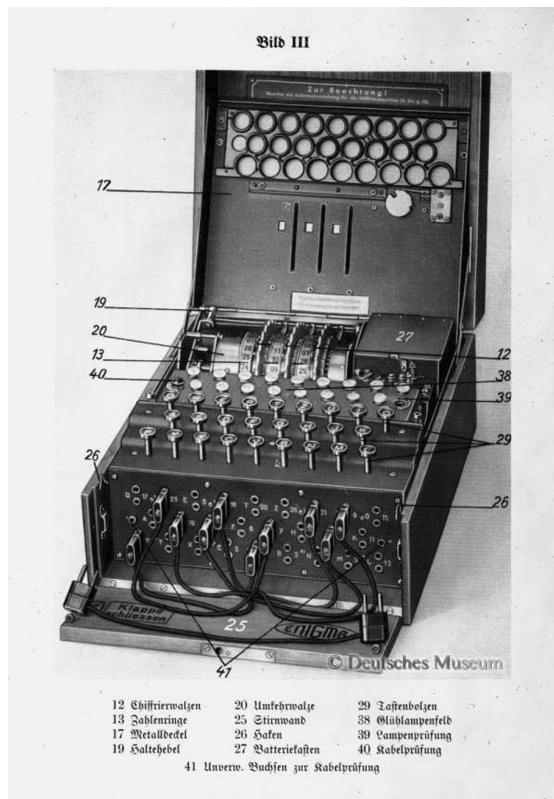


Abbildung 1.1: Die Enigma Maschine

<sup>1</sup>Eine Analogie hierfür ist der Kilometerzähler eines mechanischen Tachometers oder das Verhalten von Sekunden-, Minuten- und Stundenzeigern einer Uhr.

Bei der Enigma wurde jeden Tag eine Tagesschlüssel eingestellt, welcher durch ein Code-Buch vorgegeben war. Dieser Tagesschlüssel bestand darin, welche drei der fünf Walzen der Schlüssler in welcher Reihenfolge einzusetzen hatte. Ferner musste er die Walzenstellung einstellen. Sie bestimmt die Ausgangsstellung der Walzen. Diese konnte durch ein Sichtfenster abgelesen werden.

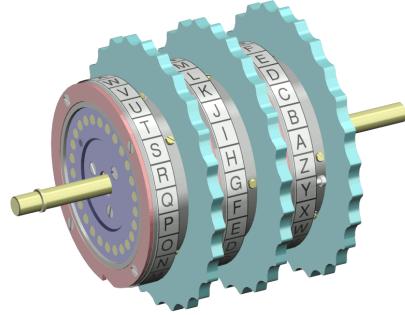


Abbildung 1.2: Enigma-Walzen

Zusätzlich musste der Schlüssler die Ringstellung des Tages einstellen. Die Ringstellung veränderte die Relation der sichtbaren Buchstaben mit der internen Verdrahtung und bewegte die Übertragskerbe (siehe Abb. 1.3), die festlegte, wann sich die Walze links von der aktuellen bewegt.

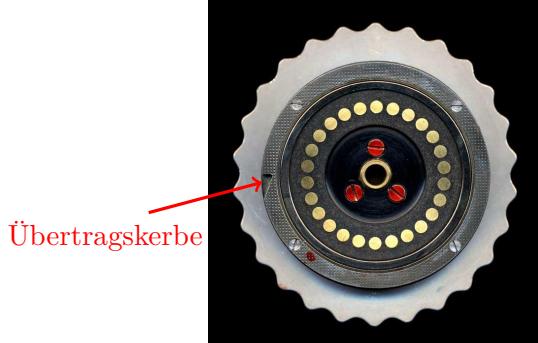


Abbildung 1.3: Walzen Frontansicht mit Übertragskerbe

Als letzte Einstellung musste das Steckerbrett verdrahtet werden. Das Steckerbrett vertauschte zwei Buchstaben miteinander. Von 13 möglichen Steckerverbindungen wurden meistens 10 vorgegeben.

Wenn der Schlüssler nun eine Taste auf der Tastatur betätigte, floss Strom durch das Steckerbrett, welches eine Substitution durchführt. Danach floss der Strom durch den Walzensatz, in den Reflektor und nochmals in entgegengesetzter Richtung durch den Walzensatz. Jeder dieser Walzen führte eine monoalphabetische Substitution durch. Nachdem der Strom den Walzensatz verlassen hatte, floss der Strom nochmals durch das Steckerbrett und erleuchtete letztendlich eine Lampe.

Abbildung 1.4: Verdrahtung Walze I

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

## 1.2 Übertragung der Nachrichten

Nachdem alle Einstellung getroffen waren, überlegte sich der Schlüssler einen „zufälligen“ Spruchschlüssel, mit dem der Text verschlüsselt wurde.<sup>2</sup> Dieser Spruchschlüssel gab die Walzenstellung für die folgende Nachricht an. Der „zufällige“ Spruchschlüssel wurde mit dem Tagesschlüssel verschlüsselt und ergab zusammen mit anderen Zusatzinformationen den „Spruchkopf“. Der Schlüssler gab nun den zu verschlüsselnden Text nach bestimmten Regeln ein. Eine Eigenschaft der Enigma ist, dass sie selbstinvers ist. Das bedeutet, dass mit der gleichen Einstellung, mit der ein Text verschlüsselt wurde, dieser auch wieder entschlüsselt werden kann.

---

<sup>2</sup>In Wahrheit wählten die Schlüssler oft den gleichen Schlüssel, der meist persönliche Informationen wie zum Beispiel den Namen der Freundin enthielt.

## 2 Bombe

### 2.1 Algorithmus Bombe

---

**Algorithm 1** Bombe Algorithmus

---

```
1: procedure BOMBE( $p_0 \dots p_{n-1} : [\text{Char}]$ ,  $c_0 \dots c_{n-1} : [\text{Char}]$ )
2:   for all rotors  $\in \text{permut(rotor order)}$ , pos  $\in [\text{AAA} \dots \text{ZZZ}]$  do
3:     plugs: Char  $\rightarrow \{\text{Char}\}$ 
4:     plugs( $p_0$ )  $\cup = \{'\text{A}'\}$ 
5:     while plugs changing do
6:       for all  $i \in [0 \dots n-1]$  do
7:         plugs( $c_i$ )  $\cup = \bigcup_{p \in \text{plugs}(p_i)} \text{encrypt}(\text{rotors}, p, \text{pos}+i)$ 
8:         plugs( $p_i$ )  $\cup = \bigcup_{p \in \text{plugs}(c_i)} \text{encrypt}(\text{rotors}, p, \text{pos}+i)$ 
9:       end for
10:      end while
11:      if  $\forall S \in \text{cod(plugs)}$ : #S < #Char then
12:        report(pos, plugs)
13:      end if
14:    end for
15:  end procedure
```

---

### 2.2 Cycle Finding Algorithm

```
typedef struct {
    char first;
    char second;
} Tuple;
```

## 3 Häufigkeitsanalyse

### 3.1 Koinzidenzindex

Zunächst benötigen wir ein geeignetes Maß, um den Grad der Annäherung eines teilweise dechiffrierten Klartextes an authentisches Deutsch abzuschätzen. Dafür verwenden wir den Koinzidenzindex:

$$IC = \frac{\sum_{i=A}^Z f_i(f_i - 1)}{N(N - 1)}$$

Sei  $f_i$  die Häufigkeit des Buchstabens in Abhängigkeit von  $i$  und  $N$  die Gesamtanzahl der Buchstaben. Wir Summieren also die Anzahl der Buchstabenpaare auf, und teilen diese durch die Anzahl der allgemein möglichen Buchstabenpaare. Der Koinzidenzindex ist also ein Maß für die Buchstabenverteilung. Für ein Text bestehend aus zufälligen Buchstaben beträgt der  $IC \approx 0.038$ ; wobei er für Deutsch  $\approx 0.076$  beträgt.

Bemerkungen:

Improvements:

Base: 8.5s

Nur letzte Buchstaben testen: 3.6s

IC: 3.3s

Low Level IO: 3.31s

Nur erster char testen an der Position, an dem das Schlagwort stehen muss: 2.25s

## **4 Quellen**