Geschwister-Scholl-Gymnasium Velbert

Die Enigmaverschlüsselung - Woran sie gescheitert ist

Mattis Jung, 2025/26
Informatik, Herr Linden

6 Wochen, bis 22. Februar 2026

Diese Facharbeit untersucht die Enigma-Verschlüsselung und die Gründe für ihr Scheitern im Zweiten Weltkrieg. Zunächst werden Aufbau und Funktionsweise der elektromechanischen Chiffriermaschine erläutert, bevor die konstruktiven und operativen Schwächen analysiert werden, die den Alliierten die Entschlüsselung ermöglichten. Im praktischen Teil wird eine eigene Enigma-Implementierung in Python vorgestellt und einer Fehleranalyse unterzogen. Abschließend wird diskutiert, wie eine hypothetische „perfekte Enigma“ die identifizierten Schwachstellen hätte beheben können.

Inhaltsverzeichnis

1 Einleitung **3**

2 Die Enigma **3**
2.1 Wie sie entstand 3
2.2 Wie sie funktioniert 4
2.2.1 Das Steckerbrett 4
2.2.2 Die Walzen 4

3 Die Schwächen der Enigma **5**
3.1 Die Umkehrwalze 5
3.2 Feste Verkabelung der Walzen 5
3.3 Kerbenmechanik 6
3.4 Das Steckerbrett 6
3.5 Fixpunktfreiheit 6
3.6 Operative Schwächen 7

4 Die Enigma Implementierung **7**
4.1 Wie ist die Implementation geplant 7
4.2 Wie funktioniert diese Implementierung? 8
4.2.1 Die Enigma - Klasse 8
4.2.2 Die Walze - Klasse 9
4.3 Was ist das Problem dieser Implementation? 10
4.3.1 Fehlende Ringstellung 11
4.3.2 Fehlender Rückwärtslauf durch die Walzen 11
4.3.3 Startposition und Kerbenlogik 11
4.3.4 Fehlende Verifikationsmöglichkeit 11

5 Die perfekte Enigma **12**
5.1 Was ist anders an dieser Enigma? 12
5.1.1 Eine verbesserte Umkehrwalze 12
5.1.2 Mehr Kerben und Double - Stepping 12
5.1.3 Austauschbare Verdrahtung 13
5.1.4 Volles Steckerbrett ohne Einschränkungen 13
5.1.5 Operative Verbesserungen 13

6 Fazit **14**

1 Einleitung

Im Zweiten Weltkrieg war die sichere Übermittlung von Nachrichten kriegsentscheidend. Die deutsche Wehrmacht setzte dafür die Enigma ein — eine elektromechanische Chiffriermaschine, die als unknackbar galt. Doch trotz ihres enormen theoretischen Schlüsselraums gelang es den Alliierten, die Enigma zu brechen. Dies lag nicht an einem einzelnen Fehler, sondern an einem Zusammenspiel aus konstruktiven Schwächen, fragwürdigen operativen Vorschriften und menschlichem Fehlverhalten.

Diese Facharbeit untersucht die Frage, woran die Enigmaverschlüsselung gescheitert ist. Dazu wird zunächst die Funktionsweise der Maschine erläutert: ihr Aufbau mit Steckerbrett, Walzen, Umkehrwalze und Ringstellung (Abschnitt 2). Anschließend werden die Schwächen analysiert, die den Kryptoanalytikern in Bletchley Park die Entschlüsselung ermöglichten — von der Fixpunktfreiheit über die feste Verkabelung bis hin zu operativen Mängeln (Abschnitt 3). Im praktischen Teil wird eine eigene Enigma-Implementierung in Python vorgestellt, deren Aufbau, Funktionsweise und Fehler beschrieben werden (Abschnitt 4). Abschließend wird diskutiert, wie eine „perfekte Enigma“ hätte aussehen können, die die identifizierten Schwachstellen behebt (Abschnitt 5).

2 Die Enigma

Die Enigma ist eine elektromechanische Chiffriermaschine. Sie verschlüsselt eine Nachricht buchstabenweise, indem sie die Zeichen über mehrere Stufen vertauscht (*Permutation*) und ersetzt (*Substitution*).

2.1 Wie sie entstand

Die Idee zur Enigma entstand bereits im Ersten Weltkrieg: Die Deutschen benötigten eine Möglichkeit, geheime Nachrichten an die Front zu übermitteln, ohne dass alliierte Truppen sie mitlesen konnten. Arthur Scherbius (1878-1929) entwickelte deshalb die Idee einer Rotor-Schlüsselmaschine und ließ sie am 23. Februar 1918 patentieren. [2] Ende 1923 erhielt die Maschine ihren Namen, als Scherbius die Erfindung nach dem griechischen Wort *enigma* („Rätsel“) benannte.

Mit der Zeit wurde die Enigma in Deutschland immer populärer. Nach dem Ende des Zweiten Weltkriegs wurden viele von den Deutschen zurückgelassene Exemplare umgebaut und weiterverwendet. Ein Beispiel ist die *Norenigma*, eine in Norwegen modifizierte Version der deutschen Enigma.

2.2 Wie sie funktioniert

Von außen erinnert die Enigma an eine Schreibmaschine. Neben der Tastatur besitzt sie jedoch ein Glühlampenfeld: Zu jeder Taste gehört eine Lampe, die beim Tastendruck den verschlüsselten Buchstaben anzeigt. Im Kern arbeitet die Enigma mit zwei entscheidenden Chiffrierelementen: dem Steckerbrett und den Walzen.

2.2.1 Das Steckerbrett

Auf dem Steckerbrett sind alle Buchstaben der Tastatur aufgeführt. Mit Steckerkabeln lassen sich jeweils zwei Buchstaben miteinander verbinden. Dadurch wird ein eingegebener Buchstabe in seinen Partnerbuchstaben umgewandelt. Es entsteht eine zusätzliche Vertauschung (*Permutation*).

Ist ein Buchstabe mit einem anderen verbunden, nennt man ihn *gesteckert*. In der Standardkonfiguration werden 10 Kabel gesteckt: 20 Buchstaben sind paarweise verbunden, die übrigen 6 bleiben ungesteckert. Zusätzlich galt die Regel, dass zwei im Alphabet direkt aufeinanderfolgende Buchstaben nicht miteinander gesteckert werden durften. Solche Vorgaben machten die Arbeit der Alliierten nicht schwieriger, sondern schränkten die möglichen Einstellungen weiter ein.

2.2.2 Die Walzen

Die Walzen besitzen auf beiden Seiten jeweils 26 Kontaktpunkte. Im Inneren ist jeder Kontaktpunkt mit genau einem anderen verdrahtet. Das bedeutet: Ein A wird nicht zwangsläufig wieder zu A, sondern beispielsweise zu E, eine *Substitution*. Werden mehrere Walzen hintereinandergeschaltet (meist drei; die *Enigma - M4* nutzte beispielsweise vier), wächst die Anzahl der möglichen Verschlüsselungen sprunghaft. Außerdem drehen sich die Walzen weiter: Nach jedem Tastendruck rotiert mindestens die rechte Walze. Erreicht sie ihre Kerbe, wird auch die nächste Walze weitergeschaltet usw.

Da die meisten Walzen nur eine Kerbe besaßen, änderte sich die linke Walze im Vergleich selten. Für den Funkverkehr der Achsenmächte nach Japan verwendeten die Deutschen jedoch eine Variante mit fünf Übertragungskerben, was diese Version gegenüber den in Deutschland am weitesten verbreiteten Enigma - Modellen stärkte (*Enigma - T*, auch *Tirpitz* genannt). Aber auch in Deutschland gab es Enigma - Varianten mit mehreren Übertragungskerben, wie die Abwehr - Enigma (G), die bis zu 17 Übert ragungskerben hatte.

3 Die Schwächen der Enigma

Durch verschiedene Regulierungen wurde die Enigma nicht kryptografisch gestärkt, sondern in entscheidenden Punkten sogar geschwächt. Diese Schwächen ergaben sich sowohl aus dem mechanischen Aufbau der Maschine als auch aus den operativen Vorschriften, die den täglichen Gebrauch regelten. Im Folgenden werden die wesentlichen Schwachstellen erläutert, die den Alliierten die Entschlüsselung ermöglichten.

3.1 Die Umkehrwalze

Die Umkehrwalze funktioniert nicht so wie die anderen Walzen, da sie nur eine Kontaktplatte hat, auf welcher die Kontakte miteinander verkabelt sind. Das führt dazu, dass wenn man ein B eintippt und ein K aufleuchtet, auch beim Eintippen vom K das B aufleuchtet. Diese Eigenschaft wird als *Involution* bezeichnet: Die Umkehrwalze erzeugt eine Abbildung, die ihre eigene Umkehrfunktion ist. Kryptografisch bedeutet dies, dass Ver- und Entschlüsselung mit denselben Einstellungen ablaufen können — ein operativer Vorteil, der jedoch zugleich die Sicherheit der Maschine erheblich minderte. Da die Umkehrwalze jeden Buchstaben stets in einen *anderen* Buchstaben überführt, kann ein Buchstabe niemals in sich selbst verschlüsselt werden (siehe auch Abschnitt 3.5). Diese Fixpunktfreiheit war eine der gravierendsten Schwächen der Enigma.

Die meisten Umkehrwalzen besaßen zudem eine feste Verkabelung, die nicht verändert werden konnte. Eine Ausnahme bildete die *UKW-D* (im alliierten Jargon *Uncle Dick* genannt), bei der die Verdrahtung vom Bediener selbst festgelegt werden konnte. Allerdings wurde diese variable Umkehrwalze erst spät eingeführt und kam nur in begrenztem Umfang zum Einsatz, sodass die meisten Enigma-Modelle weiterhin mit einer festen, den Alliierten bekannten Umkehrwalze arbeiteten.

3.2 Feste Verkabelung der Walzen

Die Verdrahtung der Rotoren war seit den 1930er Jahren festgelegt und blieb über den gesamten Kriegsverlauf unverändert. Da die Wehrmacht die Walzensätze nicht regelmäßig austauschte, konnten die Alliierten die innere Verkabelung nach und nach rekonstruieren. Dabei halfen unter anderem erbeutete Walzensätze sowie mathematische Analysen aufgefangener Funksprüche. Sobald die Verdrahtung einer Walze bekannt war, blieb sie dauerhaft kompromittiert, weil sie nicht geändert werden konnte. Nach Kerckhoffs' Prinzip hätte die Sicherheit der Enigma nicht von der Geheimhaltung der Walzenverdrahtung abhängen dürfen, sondern ausschließlich vom Schlüssel — doch in der Praxis stellte das Wissen um die Verdrahtung den Alliierten eine entscheidende Grundlage für ihre Angriffe bereit.

3.3 Kerbenmechanik

Da es in den meisten Enigma-Walzen nur eine einzige Übertragskerbe gab — eine Ausnahme bildete die Abwehr-Enigma (G), die bis zu 17 Kerben besaß — drehten sich die mittlere und die linke Walze nur selten weiter. In der Praxis bedeutete dies, dass die rechte Walze nach jedem Tastendruck rotierte, die mittlere Walze jedoch nur bei jedem 26. Tastendruck und die linke Walze nur bei jedem $26 \times 26 = 676$. Tastendruck. Dadurch blieb die linke Walze während einer typischen Nachricht (selten länger als einige hundert Zeichen) oft in derselben Position stehen. Dies reduzierte den effektiven Schlüsselraum und erlaubte es den Kryptoanalytikern in Bletchley Park, die Walzenstellungen mit deutlich weniger Aufwand einzugrenzen, als es bei gleichmäßiger Rotation aller drei Walzen der Fall gewesen wäre. Die wenigen Enigma-Varianten mit mehreren Kerben (wie die bereits erwähnte Abwehr-Enigma oder die *Enigma-T* für den Funkverkehr nach Japan) waren aus genau diesem Grund kryptografisch stärker.

3.4 Das Steckerbrett

Das Steckerbrett vertauscht Buchstaben, die miteinander gesteckert sind. Dabei durften aber nur 20 von den 26 Buchstaben gesteckert sein. Die restlichen sechs blieben ungesteckert.[1, S.108] Theoretisch hätte eine vollständige Steckerung aller 26 Buchstaben die Anzahl der möglichen Kombinationen weiter erhöht und den Angriff erschwert. Da jedoch sechs Buchstaben stets unverändert durch das Steckerbrett hindurchgingen, blieben diese als Klartext-Fragmente erhalten, was den Kryptoanalytikern wertvolle Anhaltspunkte lieferte.

Für die Alliierten und die *Codebreaker* in Bletchley Park war dies eine enorme Hilfe, da sie mit der *Bombe* (einem elektromechanischen Entschlüsselungsgerät) einige mögliche Schlüsselkombinationen sofort ausschließen konnten.[1, S.107] Die Bombe, maßgeblich von Alan Turing und Gordon Welchman entwickelt, nutzte bekannte oder vermutete Klartextfragmente (*Cribs*), um systematisch Walzenstellungen durchzuprobieren. Die ungesteckerten Buchstaben erleichterten dabei das Auffinden solcher Cribs erheblich.

Darüber hinaus galt die Vorschrift, dass zwei im Alphabet direkt aufeinanderfolgende Buchstaben nicht miteinander gesteckert werden durften. Diese Regel schränkte die möglichen Steckerkombinationen zusätzlich ein und verringerte die Sicherheit, anstatt sie zu verbessern.

3.5 Fixpunktfreiheit

Ein Buchstabe kann bei der Enigma niemals in sich selbst verschlüsselt werden. Diese Eigenschaft — die Fixpunktfreiheit — resultiert direkt aus der Konstrukti-

on der Umkehrwalze. Für die Kryptoanalytiker war sie ein mächtiges Werkzeug: Wenn ein vermutetes Klartextwort an einer bestimmten Stelle im Chiffretext den gleichen Buchstaben an der gleichen Position aufwies, konnte diese Hypothese sofort verworfen werden. So ließ sich beispielsweise das häufig verwendete Wort *WETTERBERICHT* an zahlreichen Positionen ausschließen, was die Suche nach der richtigen Zuordnung drastisch beschleunigte. Die Bombe nutzte diese Eigenschaft systematisch aus, um den Suchraum erheblich zu reduzieren.

3.6 Operative Schwächen

Zudem gab es für jeden Monat eine Schlüsseltafel, auf der Walzenlage, Ringstellung, Steckerverbindungen und Kenngruppen für jeden Tag festgelegt waren. Diese Tafeln durften nicht an Bord von Flugzeugen mitgenommen werden, da sie sonst leichter in die Hände der Alliierten fallen konnten. Dennoch gelang es den Alliierten wiederholt, solche Schlüsseltafeln zu erbeuten — beispielsweise aus versenkten U-Booten oder abgeschossenen Flugzeugen. In Kombination mit den aufgefangenen Funksprüchen ermöglichten diese erbeuteten Unterlagen den Codeknackern in Bletchley Park, die Enigma-Einstellungen zu rekonstruieren und die Entschlüsselung zu beschleunigen.

Auch menschliches Fehlverhalten schwächte die Enigma. Funker verwendeten manchmal vorhersagbare Spruchschlüssel (z. B. *AAA* oder die drei auf der Tastatur nebeneinanderliegenden Buchstaben *QWE*), was den Alliierten das Erraten erleichterte. Darüber hinaus begannen viele Nachrichten mit stereotypen Floskeln wie *AN DIE GRUPPE* oder enthielten regelmäßig wiederkehrende Wetterberichte. Solche bekannten Klartextfragmente (*Cribs*) waren für die Kryptoanalyse von unschätzbarem Wert und bildeten oft den Ausgangspunkt für einen erfolgreichen Angriff auf den Tagesschlüssel.

4 Die Enigma Implementierung

4.1 Wie ist die Implementation geplant

Der Plan der Implementation war, die Konfiguration der Enigma von ihrer Logik zu trennen. Dazu wurden mehrere JSON-Dateien erstellt, in denen die wichtigsten Informationen zur Initialisierung der Maschine gespeichert sind: Eine *vorlage.json* enthält die Tageseinstellungen (Walzenlage, Ringstellung, Steckerverbindungen und Kenngruppen), während die *rotors.json* alle verfügbaren Walzen mit ihrer Verdrahtung, ihren Kerben und ihrem Einführungsdatum auflistet.

Diese Herangehensweise orientiert sich an den historischen Schlüsseltafeln der Wehrmacht: Genau wie dort legt die Vorlage fest, welche Walzen in welcher Reihenfolge eingesetzt werden und welche Buchstaben auf dem Steckerbrett miteinander

verbunden sind. Bei der Umsetzung wurde sich jedoch zu stark auf das korrekte Einlesen und Verarbeiten dieser Daten fokussiert und die eigentliche Verschlüsselungslogik vernachlässigt. Dies fiel letztlich in den Rücken, da die Funktion der eigentlichen Maschine nie vollständig fertiggestellt wurde — die Gründe dafür werden in Abschnitt 4.3 beschrieben.

4.2 Wie funktioniert diese Implementierung?

In dieser Version der Enigma arbeiten zwei Klassen zusammen, um eine eingegebene Nachricht zu verschlüsseln: die *Enigma* - Klasse und die *Walze* - Klasse. Die Enigma - Klasse enthält die zentrale Steuerungslogik. Beim Start der Verschlüsselung erzeugt sie die benötigten Walzen - Objekte; in diesen findet die eigentliche Buchstaben - Ersetzung statt. Das Steckerbrett ist ebenfalls Teil der Enigma - Klasse und vertauscht gesteckerte Buchstaben sowohl vor als auch nach dem Durchlaufen der Rotoren.

4.2.1 Die Enigma - Klasse

Die Klasse *Enigma* wird mit drei Parametern initialisiert:

Parameter:

name (str): Name der Enigma, bestehend aus *Enigma* und der Modellbezeichnung (z. B. *I*), getrennt durch ein -. Dient zur Identifizierung des Maschinentyps (*Enigma - I, Enigma - M3, Enigma - G*, etc.). Intern wird der Name am Bindestrich aufgetrennt, um den Modelltyp zu bestimmen, der wiederum festlegt, welche Walzen aus der *rotors.json* geladen werden dürfen.

vorlage (dict): Dictionary, das alle wichtigen Informationen zu Walzenlage, Ringstellung, Steckerverbindungen und Kenngruppen enthält. Die Walzenlage ist eine Liste von Strings, die die jeweiligen Walzen spezifizieren (z. B. `[,,I‘‘, ,,IV‘‘, ,,III‘‘, ,,B‘‘]`, wobei das letzte Element die Umkehrwalze bestimmt). Die Ringstellung ist eine Liste von Zahlen, die den Rotationsversatz zwischen der inneren Verdrahtung der Walzen und den außen sichtbaren Buchstaben angibt. Die Steckerverbindung ist eine zehn Elemente lange Liste aus Strings mit je zwei Zeichen (z. B. `[,,CV‘‘, ,,WK‘‘, ...]`). Sie legt fest, welche Buchstaben miteinander gesteckert sind; dadurch bleiben sechs Buchstaben des Alphabets ungesteckert. Die Kenngruppen sind eine Liste von Strings, die mögliche Kenngruppen für den jeweiligen Tag angeben.

startingPosition (str): Ein String aus drei Zeichen (z. B. `,,QWE‘‘`), der die Grundstellung der Walzen festlegt. Beim Initialisieren wird jeder Buchstabe über `string.ascii_uppercase.index()` in seinen Zahlenwert umgewandelt und als Liste gespeichert.

Funktionen:

encode(text): Die zentrale Verschlüsselungsfunktion. Sie ruft nacheinander das Steckerbrett, die Rotoren und erneut das Steckerbrett auf und gibt den verschlüsselten Text in 5er - Gruppen formatiert zurück. Der Ablauf entspricht dem historischen Signalweg: Tastatur → Steckerbrett → Walzen → Umkehrwalze → Walzen (rückwärts) → Steckerbrett → Lampenfeld.

getFiveCharString(text): Nimmt den zusammenhängenden Chiffretext und teilt ihn in Fünfergruppen auf, getrennt durch Leerzeichen — so wie es bei der historischen Funkübertragung üblich war.

steckerSwitch(inputChar): Prüft für einen einzelnen Buchstaben, ob er in der Steckerverbindungsliste vorkommt. Falls ja, wird er durch seinen Partner ersetzt; falls nein, bleibt er unverändert. Die Funktion iteriert dazu über alle zehn Steckerpaare und vergleicht den Eingabebuchstaben mit beiden Zeichen jedes Paares.

steckerLoop(text): Wendet *steckerSwitch* auf jeden Buchstaben eines gesamten Textes an. Vorher werden Leerzeichen entfernt und alle Buchstaben in Großbuchstaben umgewandelt.

rotors(text): Führt die eigentliche Walzenverschlüsselung durch. Zunächst werden die Walzen über *getRotors* geladen. Dann wird für jeden Buchstaben der Nachricht der Signalweg durch die Walzen simuliert. Die Durchlaufreihenfolge ist als Liste `[0, 2, 3, 4, 1, 4, 3, 2, 0]` definiert, wobei Index 0 die Eintrittswalze (ETW), Index 1 die Umkehrwalze (UKW) und die Indizes 2–4 die drei Rotoren darstellen. Das Signal durchläuft also: ETW → Rotor 1 → Rotor 2 → Rotor 3 → UKW → Rotor 3 → Rotor 2 → Rotor 1 → ETW. Nach jedem verschlüsselten Buchstaben wird *rotate* aufgerufen.

rotate(): Implementiert die Kerbenmechanik. Die rechte Walze (Index 2) rotiert bei jedem Tastendruck. Gibt ihre *rotate* - Methode *True* zurück (d. h. sie hat ihre Kerbe erreicht), wird auch die mittlere Walze (Index 3) weitergedreht, und so fort.

getRotors(referencePath, startingPositions): Lädt die Datei *rotors.json* und erzeugt die benötigten Walzen - Objekte. Für Enigma - Modelle vom Typ *I*, *M3* oder *M4* werden ETW, UKW und die drei (bzw. vier) Rotoren aus der JSON - Datei gesucht und als *Walze* - Objekte instanziiert. Die Startpositionen aus der Grundstellung werden dabei an die jeweiligen Rotoren übergeben.

4.2.2 Die Walze - Klasse

Die Klasse *Walze* repräsentiert eine einzelne Walze (Rotor, Eintrittswalze oder Umkehrwalze) und wird mit folgenden Parametern initialisiert:

Parameter:

source (dict): Ein Dictionary aus der *rotors.json* mit den Schlüsseln `walze` (Name, z. B. „III“), `wiring` (Verdrahtung als 26 - Zeichen - String), optional `kerbe` (Kerbenbuchstabe) und `modell` (Liste kompatibler Enigma - Modelle).

start (int) = 0: Die Startposition der Walze als ganzzahliger Wert (0–25), welcher der Grundstellung entspricht. Standardmäßig steht die Walze auf Position 0 (Buchstabe *A*).

Attribute:

walzenart (str): Der Name der Walze (z. B. ,,III‘‘, ,,UKW B‘‘ oder ,,ETW‘‘).

wiring (str): Die Verdrahtung als 26 - Zeichen - String, der angibt, welcher Buchstabe auf welchen abgebildet wird.

encoding (list[int]): Eine aus dem *wiring* - String berechnete Liste von 26 ganzzahligen Verschiebungswerten. Für jede Position i wird die Differenz zwischen dem Zielindex und i gespeichert. Dadurch kann die Verschlüsselung eines Buchstabens als einfache Addition und Modulooperation durchgeführt werden, anstatt den Buchstaben im String zu suchen.

kerbe: Der Kerbenbuchstabe (als Zahlenwert) bei normalen Rotoren, *None* bei ETW und UKW.

rotierbar (bool): Gibt an, ob sich die Walze drehen kann. ETW und UKW sind nicht rotierbar.

position (int): Die aktuelle Position der Walze.

Funktionen:

getEncoding(): Berechnet die *encoding* - Liste aus dem *wiring* - String. Für jeden Buchstaben an Position i im Wiring wird die Verschiebung Index des Buchstabens$-i$ berechnet.

rotate(): Erhöht die Position um 1 (modulo 26). Gibt *True* zurück, wenn die Walze nach der Rotation ihre Kerbe erreicht hat — das Signal für die nächste Walze, ebenfalls zu rotieren.

run(item): Führt die Verschlüsselung eines einzelnen Buchstabens (als Ganzzahl 0–25) durch. Die Berechnung lautet:

$$\text{retItem} = (\text{item} + \text{encoding}[(\text{item} + \text{position}) \bmod 26]) \bmod 26$$

Die aktuelle Position der Walze wird dabei berücksichtigt, sodass die Verschlüsselung von der Walzenstellung abhängt.

4.3 Was ist das Problem dieser Implementation?

Das grundlegende Ziel der Enigma - Implementierung ist es, Nachrichten zu verschlüsseln, sodass sie ohne den korrekten Schlüssel nicht lesbar sind. Da man einen bestimmten Schlüssel (Walzenlage, Grundstellung, Ringstellung, Steckerverbindungen) benötigt, ist die Verifikation schwierig: Es ist nicht ohne Weiteres erkennbar, ob ein verschlüsselter Text das korrekte Ergebnis darstellt oder nur zufällig aussehenden Buchstabensalat — beides sieht auf den ersten Blick gleich aus. Im Folgenden werden die identifizierten Fehler und Probleme dieser Implementation analysiert.

4.3.1 Fehlende Ringstellung

Die Ringstellung wird zwar aus der *vorlage.json* geladen und im Attribut `self.ringstellung` gespeichert, jedoch nirgendwo in der Verschlüsselungslogik verwendet. Bei der historischen Enigma verschiebt die Ringstellung die Zuordnung zwischen den äußeren Buchstabenringen und der inneren Verdrahtung. Ohne Berücksichtigung der Ringstellung produziert die Implementierung falsche Ergebnisse, selbst wenn alle anderen Einstellungen korrekt sind. Die Ringstellung hätte in der Klasse *Walze* beim Initialisieren als zusätzlicher Versatz in die *encoding* - Berechnung oder in die *run* - Methode einfließen müssen.

4.3.2 Fehlender Rückwärtslauf durch die Walzen

In der historischen Enigma durchläuft das Signal die Walzen auf dem Hinweg in die eine Richtung und auf dem Rückweg (nach der Umkehrwalze) in die entgegengesetzte Richtung. Diese Implementierung verwendet jedoch dieselbe *run* - Methode für beide Richtungen. Die *encoding* - Liste bildet nur den Vorwärtslauf ab, d. h. für jeden Eingangskontakt wird der Ausgangskontakt berechnet. Für den Rückweg müsste die inverse Abbildung verwendet werden: gegeben der Ausgangskontakt, welcher Eingangskontakt gehört dazu? Da dies nicht implementiert ist, ist der Rückweg durch die Walzen fehlerhaft und die Verschlüsselung insgesamt inkorrekt.

4.3.3 Startposition und Kerbenlogik

Die Startpositionen werden als Liste `[0, 1, 2]` an *getRotors* übergeben und den Rotoren über den Index der *walzenlage* - Liste zugeordnet. Allerdings enthält *walzenlage* auch die Bezeichnung der Umkehrwalze als letztes Element. Die Startpositionen beziehen sich jedoch nur auf die drei Rotoren, nicht auf die Umkehrwalze. Dadurch kann es zu einer Indexverschiebung kommen.

Zudem prüft die *rotate* - Methode, ob die Position gleich der Kerbe ist (`self.position == self.kerbe`). Die Rotation findet jedoch *vor* dem Vergleich statt. Bei der historischen Enigma existiert außerdem der sogenannte *Double - Stepping* - Mechanismus: Wenn die mittlere Walze ihre Kerbe erreicht, wird sie beim nächsten Tastendruck *erneut* weitergedreht, zusammen mit der linken Walze. Dieser Mechanismus ist in dieser Implementierung nicht berücksichtigt.

4.3.4 Fehlende Verifikationsmöglichkeit

Da die Verschlüsselung aufgrund der oben genannten Fehler keine korrekten Ergebnisse liefert, fehlt auch eine Möglichkeit, die Ergebnisse zu verifizieren. Eine korrekt implementierte Enigma muss die Eigenschaft besitzen, dass eine Nachricht,

die mit denselben Einstellungen zweimal verschlüsselt wird, wieder den ursprünglichen Klartext ergibt (da die Enigma eine Involution ist). Diese Eigenschaft hätte als grundlegender Test verwendet werden können: wenn `encode(encode(text)) == text`, ist die Implementation zumindest in sich konsistent. Da dies bei dieser Implementierung nicht der Fall ist, wurde deutlich, dass fundamentale Fehler in der Verschlüsselungslogik vorhanden sind.

5 Die perfekte Enigma

Die Analyse der Schwächen der historischen Enigma wirft eine naheliegende Frage auf: Wie hätte eine Enigma aussehen müssen, die mit den damals verfügbaren Mitteln nicht zu knacken gewesen wäre? In diesem Abschnitt wird ein hypothetisches Redesign der Enigma skizziert, das die in Abschnitt 3 beschriebenen Schwachstellen systematisch behebt.

5.1 Was ist anders an dieser Enigma?

Die „perfekte Enigma“ unterscheidet sich in mehreren entscheidenden Punkten von der historischen Maschine:

5.1.1 Eine verbesserte Umkehrwalze

Die gravierendste Schwäche der historischen Enigma — die Fixpunktfreiheit — ließe sich durch eine Umkehrwalze beseitigen, die auch Fixpunkte zulässt, d. h. einen Buchstaben auf sich selbst abbilden kann. Alternativ könnte die Umkehrwalze ganz entfallen und durch eine vierte (oder fünfte) reguläre Walze ersetzt werden. Dies würde allerdings bedeuten, dass Verschlüsselung und Entschlüsselung nicht mehr mit denselben Einstellungen ablaufen können — eine Einschränkung, die aus operativer Sicht nachteilig wäre, die Sicherheit jedoch erheblich steigern würde. Ein Mittelweg wäre die konsequente Verwendung der variablen Umkehrwalze $UKW\text{-}D$, deren Verdrahtung täglich gewechselt wird. Allein dadurch würde der Schlüsselraum um einen signifikanten Faktor wachsen.

5.1.2 Mehr Kerben und Double-Stepping

Die perfekte Enigma würde Walzen mit mehreren, ungleichmäßig verteilten Kerben verwenden — ähnlich der Abwehr-Enigma mit ihren bis zu 17 Kerben. Dadurch würden sich alle Walzen deutlich häufiger drehen und die effektive Periodenlänge der Verschlüsselung erheblich verlängern. Die linke Walze, die bei der historischen Enigma oft während einer gesamten Nachricht in derselben Position blieb, würde

nun regelmäßig weiterschalten. Dies würde die Kryptoanalyse erschweren, da die Walzenstellungen nicht mehr so leicht eingegrenzt werden könnten.

5.1.3 Austauschbare Verdrahtung

Anstelle einer festen, werksseitig eingebauten Verdrahtung sollten die Walzen eine austauschbare oder vom Bediener konfigurierbare Verdrahtung besitzen. In der Praxis wäre dies durch herausnehmbare Verdrahtungseinsätze realisierbar, die regelmäßig gewechselt werden. Selbst wenn dem Gegner einzelne Einsätze in die Hände fielen, wären nur die Nachrichten des jeweiligen Zeitraums kompromittiert — nicht der gesamte vergangene und zukünftige Funkverkehr.

5.1.4 Volles Steckerbrett ohne Einschränkungen

Die perfekte Enigma würde alle 26 Buchstaben auf dem Steckerbrett steckern, sodass kein Buchstabe unverändert hindurchgeht. Darüber hinaus würde die Regel entfallen, dass aufeinanderfolgende Buchstaben nicht miteinander gesteckert werden dürfen. Beide Einschränkungen verringerten den Schlüsselraum unnötig und lieferten den Kryptoanalytikern wertvolle Informationen.

5.1.5 Operative Verbesserungen

Neben den mechanischen Änderungen wären auch operative Maßnahmen entscheidend. Spruchschlüssel sollten nicht vom Funker frei gewählt, sondern durch ein Zufallsverfahren erzeugt werden, um vorhersagbare Schlüssel wie *AAA* oder *QWE* auszuschließen. Standardfloskeln am Anfang und Ende von Nachrichten — wie *AN DIE GRUPPE* oder *WETTERBERICHT* — müssten verboten oder zumindest systematisch variiert werden, um den Alliierten keine Cribs zu liefern. Außerdem sollte die maximale Nachrichtenlänge strikt begrenzt werden, um die Menge an Chiffretext zu reduzieren, die mit demselben Schlüssel erzeugt wird.

6 Fazit

Die Enigma war für ihre Zeit eine bemerkenswert komplexe Chiffriermaschine. Mit ihren austauschbaren Walzen, dem Steckerbrett und der sich bei jedem Tastendruck ändernden Verschlüsselung bot sie einen theoretischen Schlüsselraum, der mit den damaligen Mitteln kaum vollständig durchsuchbar schien. Dennoch wurde sie geknackt — nicht weil die Grundidee fehlerhaft war, sondern weil eine Kombination aus konstruktiven Entscheidungen, operativen Vorschriften und menschlichem Fehlverhalten die Sicherheit systematisch untergrub.

Die Fixpunktfreiheit, die feste Verkabelung, die unzureichende Kerbenmechanik und die unvollständige Steckerung boten den Kryptoanalytikern in Bletchley Park Angriffsflächen, die sich mit der *Bombe* effizient ausnutzen ließen. Hinzu kamen operative Schwächen wie vorhersagbare Spruchschlüssel und stereotype Nachrichtenanfänge, die den Alliierten die entscheidenden Cribs lieferten.

Die eigene Enigma - Implementierung hat gezeigt, wie anspruchsvoll selbst eine vereinfachte Nachbildung dieser Maschine ist. Fehlende Ringstellung, ein nicht implementierter Rückwärtslauf durch die Walzen und eine unvollständige Kerbenmechanik führten dazu, dass die Simulation keine korrekten Ergebnisse liefert. Dies verdeutlicht, wie eng die korrekte Verschlüsselung an das präzise Zusammenspiel aller Komponenten gebunden ist.

Die Überlegungen zur „perfekten Enigma“ zeigen, dass viele der Schwächen mit vergleichsweise einfachen Maßnahmen hätten behoben werden können: eine variable Umkehrwalze, mehr Kerben, austauschbare Verdrahtungen und strengere operative Vorschriften. Dass diese Maßnahmen nicht umgesetzt wurden, lag oft an der Priorisierung operativer Bequemlichkeit gegenüber kryptografischer Sicherheit.

Abschließend lässt sich feststellen, dass die Geschichte der Enigma eine zeitlose Lehre enthält: Die Sicherheit eines Verschlüsselungssystems ist stets nur so stark wie sein schwächstes Glied — und dieses Glied ist häufig nicht die Mathematik, sondern der Mensch.

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Facharbeit selbstständig angefertigt, keine anderen als die angegebenen Hilfsmittel benutzt und die Stellen der Facharbeit, die im Wortlaut oder im wesentlichen Inhalt von anderen Autoren übernommen wurden, mit genauer Quellenangabe kenntlich gemacht habe.

Velbert, den 22. Februar 2026

Mattis Jung

Literatur

[1] Francis Harry Hinsley und Alan Stripp. *Codebreakers: the inside story of Bletchley Park*. Oxford University Press, 2001.

[2] *Patentschrift Chiffrierapparat DRP Nr. 416 219*. URL: `https://www.cdvandt.org/Enigma%20DE416219C1.pdf` (besucht am 06. 02. 2026).