7 Wehrmacht Enigma and Known Plaintext

The Polish break into the Enigma relies on the way in which the German operators handled the message keys. With the beginning of the war the method of message keying changed and the pre-war cryptanalytic approaches broke down.

Equations for Known Plaintext

Already the Polish cryptanalysts had exlored the idea of using known plaintext—starting from the observation that the German military in their messages used a lot of stereotypical phrases such as "Heil Hitler" or "Oberkommando der Wehrmacht" (= Army's High Command). Chunks of known plaintext (called "cribs" by the british cryptanalysts) allow narrowing down the exhaustive search to an amount that eventually may be mastered with the help of some cleverly constructed electro-mechanical machines. Alan Turing largely and systematically expanded this approach.

Here is an example (Example 1, taken from [2] as virtually all authors of cryptographic texts do). Let the ciphertext

```
ULOEB ZMGER FEWML KMTAW XTSWV UINZP R ...
```

be given. We suppose the message contains the phrase "Oberkommando der Wehrmacht" near the beginning. A negative pattern search over the first 12 possible positions yields exactly one hit:

```
U L O E B Z M G E R F E W M L K M T A W X T S W V U I N Z P R
o b e r k o = m a n d o d e r w e h r m a c h t
  o b = r k o m m a n d o d e r w e h r m a c h t
    = b e r k o m m a n d o d e r w e h r m a c h t
      o = e r k o m m a n d o d e r w e h r m a c h t
        o b e r k o m m a n d o d e r = e h r m a c h t
===>      o b e r k o m m a n d o d e r w e h r m a c h t
            o b = = k o m = a n d o d e r w e h r m a c h t
              o b e r k o = m a n d o d e r w e h r m a c h t
                o b e r k o m m a n d o d e r = e h r m a c h
                  o b = r k o m = a n d o d e r w e h r m a c
                    o b e r k o = m = n d o d e r w e h r m a
                      o b e r = o m m a n d o d e r w e h r m
```

We assume the rotor wirings of all five rotors as known. The naive approach—exhaustion by brute force and assuming that the ring settings don't interfere with the crib—would go through all 60 possible rotor orders, all $26^3 = 17576$ start positions, and all $> 10^{14}$ plug configurations, each time decrypt the ciphertext, and look if the known plaintext results. The huge number of plug configurations makes this approach hopeless, the "virtual"

keylength for this approach being about 67 bits ($10^{23}/26^2 \approx 1.6 \cdot 10^{20} \approx 2^{67}$). (We first neglect the ring settings that have little impact on the cryptanalysis.)

Fortunately, using known plaintext, we may find conditions that involve *only a single plug*. Recall the general situation as shown in Figure 4.

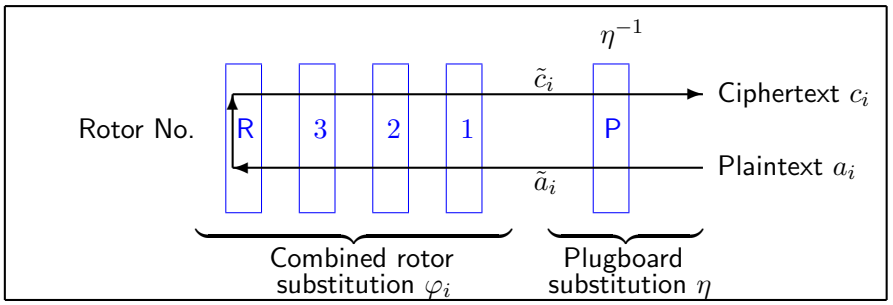

Figure 4: *Enigma with plugboard*

Assume a sequence $a_1 \dots a_m$ of known plaintext is given with corresponding ciphertext $c_1 \dots c_m$, the respective combined rotor substitutions being $\varphi_1, \dots, \varphi_m$ and the "full" Enigma substitutions, $\rho_i = \eta^{-1}\varphi_i\eta$. This gives the equations

$$\begin{aligned} c_1 = \rho_1 a_1 &= \eta^{-1}\,\varphi_1\,\eta\,a_1 \\ &\vdots \\ c_m = \rho_m a_m &= \eta^{-1}\,\varphi_m\,\eta\,a_m \end{aligned}$$

or $\eta c_i = \varphi_i \eta a_i$. Denoting the image of a letter under the (fixed but unknown) plugboard substitution by a tilde we get:

Lemma 1 *For a sequence $a_1 \dots a_m$ of known plaintext we have*

$$\tilde{c}_i = \varphi_i\,\tilde{a}_i \quad \textit{and} \quad \tilde{a}_i = \varphi_i\,\tilde{c}_i \quad \textit{for } i = 1, \dots, m.$$

For the second equation we used the fact that the combined rotor substitutions φ_i are involutions.

Looking for Cycles

Returning to Example 1 we consider the special situation

```
                          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2
i   = 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
c_i = Z M G E R F E W M L K M T A W X T S W V U I N Z
a_i = O B E R K O M M A N D O D E R W E H R M A C H T
```

From such a plaintext-ciphertext pair we extract the TURING **graph**: The nodes correspond to the letters A ... Z of the standard alphabet. For each pair (a_i, c_i) of plaintext letter and corresponding ciphertext letter an edge is drawn between these two letters, and this edge is labeled by the index i. Due to the reciprocity between plaintext and ciphertext, the situation is modeled by an undirected graph. An edge with label j between nodes s and t means that $t = \rho_j s$ and $s = \rho_j t$—or $\tilde{t} = \varphi_j \tilde{s}$ and $\tilde{s} = \varphi_j \tilde{t}$. Figure 5 shows the TURING graph for Example 1.

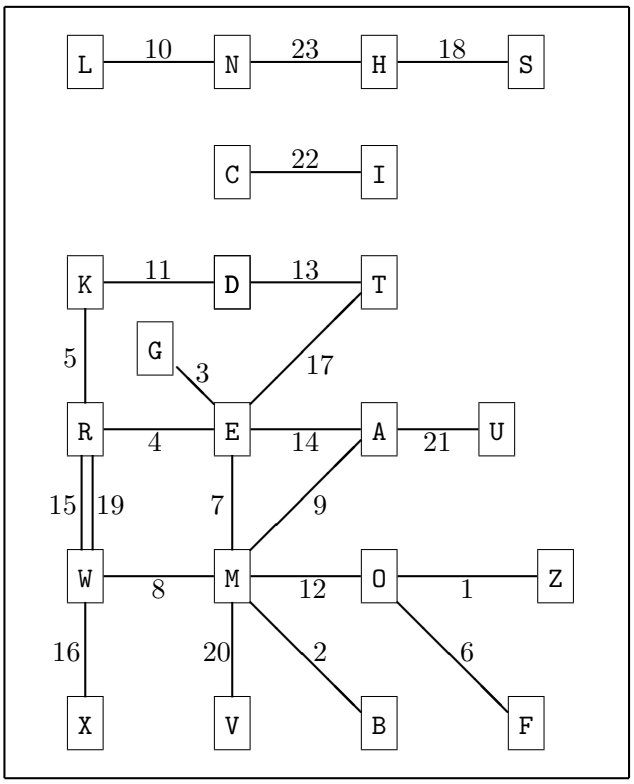

Figure 5: TURING *graph for Example 1*

TURINGs approach uses the cycles in this graph ("closures" in TURING's way of speaking). In the notation of Lemma 1 we find:

$$\mathtt{E} = \rho_7\,\mathtt{M}, \quad \mathtt{M} = \rho_9\,\mathtt{A}, \quad \mathtt{A} = \rho_{14}\,\mathtt{E}, \quad \text{and} \quad \tilde{\mathtt{E}} = \varphi_7\,\tilde{\mathtt{M}}, \quad \tilde{\mathtt{M}} = \varphi_9\,\tilde{\mathtt{A}}, \quad \tilde{\mathtt{A}} = \varphi_{14}\,\tilde{\mathtt{E}},$$

and combine these three equations into one **cycle equation**

$$\mathtt{E} = \rho_7\,\rho_9\,\rho_{14}\,\mathtt{E}. \quad \text{and} \quad \tilde{\mathtt{E}} = \varphi_7\,\varphi_9\,\varphi_{14}\,\tilde{\mathtt{E}}.$$

In general we have:

Theorem 2 (Fixed Point Theorem of Rejewski/Turing**)** *Let ρ_i be the Enigma substitution in position i, and $\varphi_i = \eta\rho_i\eta^{-1}$ be the substitution without plugs. Then a letter a is a fixed point of a composition $\rho_{i_1} \cdots \rho_{i_k}$ if and only if the plugged letter $\tilde{a}$ is a fixed point of $\varphi_{i_1} \cdots \varphi_{i_k}$.*

Thus the fixed point property of a cycle is in a certain sense independent of the plug connections.

Corollary 1 (Turing**'s cycle condition)** *Each loop in the* Turing *graph gives a necessary condition for the correct key of the Enigma encryption in the form*

$$\tilde{a} = \varphi_{i_1} \ldots \varphi_{i_k}\, \tilde{a}$$

for a letter a. In particular $\tilde{a}$ is a fixed point of the corresponding composition of unplugged Enigma substitutions.

Although mathematically trivial this theorem and its corollary are the keys to eliminating the complexity of the plugboard by a meet-in-the-middle attack.

What is the benefit of Turing's cycle condition? Suppose in Example 1 we try all 26 possible values for $\tilde{\mathtt{E}} = \eta\,\mathtt{E}$ and all 26^3 possible rotor positions for all 60 possible rotor orders, searching for fixed points of $\varphi_7\,\varphi_9\,\varphi_{14}$—an exhaustion of $60 \times 26^4 = 27,418,560$ cases. Then the probability that the cycle condition is fulfilled is about $1/26$. This rules out $\approx 25/26 \approx 96\%$ of all cases and leaves us with $\approx 60 \times 26^3$ cases—not really impressive, but it could be a good start: Suppose we find two cycles involving $\mathtt{E}$, then we are left with $\approx 60 \times 26^2$ cases, for three cycles with $\approx 60 \times 26$ cases, for four cycles with ≈ 60 cases, i. e. with the exhaustion of the possible rotor orders. And the outcome of this search is:

- The correct initial rotor positions for our known plaintext
- The correct plugboard images for all letters that occur in one of the cycles—a significant part of the complete plug configuration

Now in our Example 1 (that is in fact Deavour's and Kruh's) we see two other cycles involving $\mathtt{E}$:

$$\tilde{\mathtt{E}} = \varphi_4\,\tilde{\mathtt{R}}, \quad \tilde{\mathtt{R}} = \varphi_{15}\,\tilde{\mathtt{W}}, \quad \tilde{\mathtt{W}} = \varphi_8\,\tilde{\mathtt{M}}, \quad \tilde{\mathtt{M}} = \varphi_7\,\tilde{\mathtt{E}},$$

and

$$\tilde{\mathtt{E}} = \varphi_4\,\tilde{\mathtt{R}}, \quad \tilde{\mathtt{R}} = \varphi_5\,\tilde{\mathtt{K}}, \quad \tilde{\mathtt{K}} = \varphi_{11}\,\tilde{\mathtt{D}}, \quad \tilde{\mathtt{D}} = \varphi_{13}\,\tilde{\mathtt{T}}, \quad \tilde{\mathtt{T}} = \varphi_{17}\,\tilde{\mathtt{E}},$$

giving the two additional cycle conditions

$$\tilde{\mathtt{E}} = \varphi_4\,\varphi_{15}\,\varphi_8\,\varphi_7\,\tilde{\mathtt{E}}, \quad \tilde{\mathtt{E}} = \varphi_4\,\varphi_5\,\varphi_{11}\,\varphi_{13}\,\varphi_{17}\,\tilde{\mathtt{E}}.$$

The complete cycle constellation may be visualized by Figure 6.

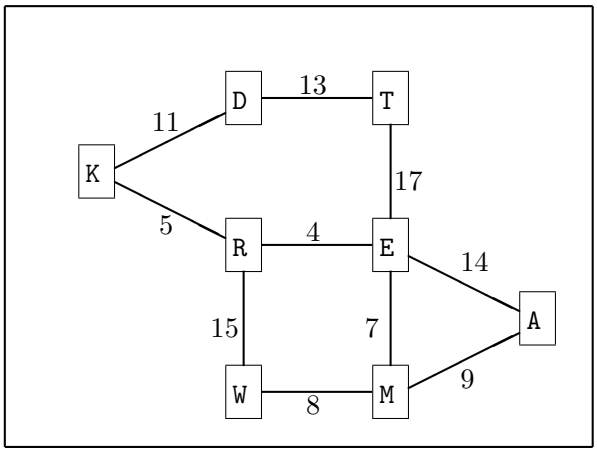

Figure 6: TURING *cycles in Example 1*

Evaluating the Cycle Conditions

In evaluating the cycle conditions one sets the rotors to start positions and then steps Rotor 1 only. In lucky cases also in the real situation only Rotor 1 moves. In bad cases Rotor 2 moves, maybe even Rotor 3. Since the ring setting is unknown, these stepping positions are unknown. Because in the example all the cycles are between plaintext positions 4 and 17, the length of the effectively used plaintext segment is 14, and the probability for a stepping of Rotor 2 in between is $13/26 = 50\%$, a stepping that would invalidate the approach, and a good argument for using rather short cribs.

Now assume that we have identified the correct rotor order and the correct initial positions of all the rotors, and no interfering movement of Rotors 2 and 3 occurs for the involved plaintext section $a_1 \dots a_m$. Then the combined rotor substitutions $\varphi_1, \dots, \varphi_m$ are known, and the plug image $\tilde{s} = \eta s$ is known for all letters s that occur in the cycles. In the example we know $\tilde{\mathtt{E}} = \eta\mathtt{E}$ and consequently

$$\tilde{\mathtt{R}} = \varphi_4\tilde{\mathtt{E}}, \quad \tilde{\mathtt{K}} = \varphi_5\tilde{\mathtt{R}}, \quad \tilde{\mathtt{M}} = \varphi_7\tilde{\mathtt{E}}, \quad \tilde{\mathtt{W}} = \varphi_8\tilde{\mathtt{M}}, \quad \tilde{\mathtt{A}} = \varphi_9\tilde{\mathtt{M}},$$

$$\tilde{\mathtt{D}} = \varphi_{11}\tilde{\mathtt{K}}, \quad \tilde{\mathtt{O}} = \varphi_{12}\tilde{\mathtt{M}}, \quad \tilde{\mathtt{T}} = \varphi_{13}\tilde{\mathtt{D}}, \quad \tilde{\mathtt{X}} = \varphi_{16}\tilde{\mathtt{W}}.$$

Furthermore we find $\tilde{\mathtt{F}} = \varphi_6\tilde{\mathtt{O}}$. Since η is an involution the inverse relations might involve further letters. That is we know the plugboard substitutes of at least 11 letters.

What is yet missing is

- The plugboard substitutes of the remaining letters
- The stepping position of Rotor 2

To continue assume first that the remaining letters are unchanged by the plugboard and decrypt $c_{m+1}, \ldots$ As soon as the resulting plaintext is unreadable either a new plugboard connection or the stepping position is detected. If the crib occurred in the middle of the ciphertext, we run the same procedure backwards to the beginning of the message.

Conclusion

The huge number of possible plug settings turns out to be an illusory complication: The exhaustion used the plug connection of a single letter only. In good cases where the procedure yields a unique solution of the cycle conditions the effort was testing 26 plug connections with 26^3 start positions for each of the 60 rotor orders, that is $27,418,560 \approx 1.6 \cdot 2^{24}$ cases. In each case we have to do some trial encryptions for the letters in the cycles plus some house-keeping plus some finishing. So we may guess that the search space is dropped to about 30 bits.

As soon as the daily key—rotor order, ring settings, plug connections, initial positions of the rotors—is known, reading all further messages of the day comes for almost no additional costs because all message keys are encrypted with the same initial rotor positions.

A Note on the Technical Realization: Turing's Bombe

Turing's Bombe consisted of a battery of several Enigmas (without plugboards), called "scramblers" and in one-to-one correspondence with the nodes of the Turing graph, synchronously stepping through all 26^3 rotor positions. For each edge two scramblers were connected by a cable, and set to start positions differing by the number that corresponded to the label of the edge. Therefore the physical arrangement of the components was an exact model of the graph. The cable had 26 wires, so all choices for the plug connection of a selected letter ($\tilde{\mathrm{E}}$ in Example 1) could be tested in parallel. The cycle conditions corresponded to closed electrical circuits that made the bombe stop. Then the operator noted the actual rotor positions and restarted the bombe with the next set of positions.

Using enough scramblers even all the sixty rotor orders could be tested in parallel, dropping the effective search costs to 26^3, equivalent with a complexity of 14 bits only. A complete run of the bombe took 11 minutes. (Today a simulation on a PC without parallel execution takes about 5 minutes.)

Unfortunately in general the solution was far from unique, so the bombe produced a huge number of "false positive" stops. An idea of Welchman largely reduced the number of false positives by a clever add-on to the bombe, see Section 8 below, and this was crucial for the success of the British cryptanalysts against the Enigma.