GEOMETRY

Lecture 4

CONVENTIONAL CRYPTOGRAPHY

by

PROFESSOR HAROLD THIMBLEBY
Gresham Professor of Geometry

21 February 2002

The reduced Enigma project

Harold Thimbleby
Gresham Professor of Geometry

Conventional Cryptography, held at Gresham College on 21 February 2002, is the first of two lectures on cryptography. The second will be *Modern Cryptography*, to be held on 28 February, also at Gresham College. These draft notes accompany the first lecture, *Conventional Cryptography*. Additional notes will be provided at the second lecture, covering the material in both lectures.

Cryptography is a fascinating subject. The paradoxical concept of sharing secrets has a natural appeal, and can be used as a motivation to discuss mathematics and computer science, or to help introduce more advanced concepts to audiences: such as public key cryptography and digital signatures. These notes discuss the design of a fully operational, simplified cryptographic machine, based closely on the Enigma, designed for hands-on use in lectures. The machine is interesting not just because it makes the rather complex operation of the Enigma much clearer, but because it exposes some of the fundamental design problems of the Enigma in a new way. (Elsewhere we have discussed how to introduce the underlying concepts to audiences of any age or skill, from school children to postgraduate specialists: see next week's notes.)

Alfred Scherbius was one of many people in the early 1900s to patent the basic idea of a mechanical or electromechanical code machine based on rotors. There were many designs and developments, but the most important of these devices was the Enigma, as used extensively by the Germans during the Second World War. A film *Enigma* has been made about breaking the code; excellent written accounts are Simon Singh's popular book *The Code Book*, and David Kahn's *The Codebreakers* which is a standard reference. A mathematician's introduction to the field is Freidrich Bauer's *Decrypted Secrets*.

Scherbius's Enigma had a 26 button keyboard (with no space button) and 26 corresponding light bulbs. Briefly, a plugboard and a set of rotors mess up the wiring, so that as buttons are pressed one of the 26 light bulbs come on in turn, but not in any obvious order. The plugboard settings and initial rotor settings represent the key, and another Enigma with the same settings can decode the message.

Original WWII Enigmas are now valuable antiques, and rebuilding an accurate Enigma is both a complex job, and one that creates a complex code that is hard to explain in lectures. What, then, is the simplest Enigma that can be made?

One might think that the simplest Enigma would have two buttons, say labelled in binary as 0 and 1. Two light bulbs would also be labelled 0 and 1. When an operator types out a binary message, say, 01011001... the lights would perhaps code it as 11001010... This is clearly the simplest scheme inspired by the Enigma. Surprisingly, an Engima with only two buttons would have a disastrous property: 0 would always code as 1, and 1 always as 0. This is such a simple substitution code it hardly merits being called a code! The problem arises due to the Enigma's wiring, but is usually explained differently: no letter can code as itself. This property of the Enigma was one of its main weaknesses, and enabled many coded transmissions to be broken. Exploiting this weakness, the British cryptanalysts at Bletchley Park built a machine, the Bombe, that automatically looked for possible matches.

Figure 1 shows an Enigma circuit taken from a German navy manual. It is easiest to follow the wiring from the battery connection +. Suppose the Y button is pressed. Current is connected (along the solid wire at the far left) up to the rotors. The three rotors make a 'random' connection through to the reflector, and then back, where the circuit continues in the dashed wire (emerging from the rotors at the top right). The return wire connects (in this case) to the Z button, where it is connected through to the Z lamp. All lamps have a common return to the – battery connection. So for these rotor positions, if Y is pressed Z lights; and if Z is pressed Y lights.

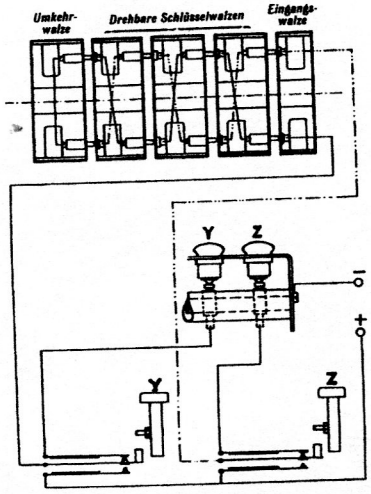

Figure 1. Part of the Enigma circuit, from a German navy manual (from Kahn, 1991b).

In actual operation, when a button is pressed the rotors move, and connection of Y→Z and Z→Y would not be repeated for a long time. Of course if Y and Z are the only buttons, they cannot be connected any other way than this, however the rotors work. The German navy manual shows a degenerate Enigma!

The Engima's weakness: no letter can code as itself facilitated code breaking. If a crib (a suspected word in the plaintext) is slid along a ciphertext, if no letters correspond anywhere, then the crib is a possible decrypt at that point. A good crib might be HEILHITLER, not least because it was said often, but because it has a pattern of several Hs and Es and Is. A Bombe would typically find several possible Enigma settings that could possibly code a message containing HEILHITLER, and a room full of Enigma operators would work through the possible settings systematically until the message was successfully decoded.

One might suppose that had Scherbius made a simplified Enigma first, he would have spotted the problem. Instead he went for 26 letters straight away, and hence disguised the flaw in a layer of complexity. He fell for the classic *complication illusoire*. Making codes more complex rarely makes them harder to solve.

Since every letter is coded as another letter, it might seem possible to build a simplified Enigma with three buttons and three lights. Suppose the buttons were XYZ. There are just two possible codes: X→Y, Y→Z, Z→X or X→Z, Y→X, Z→Y.

But the Enigma was designed to be reciprocal, so that, with the same key settings, it could be used to both code and decode messages. With three keys this is not possible. Suppose X codes as Y, then at the same setting, Y would code to Z, not to X. Instead, the Enigma should code so that if X codes as Y, Y codes as X. This is true for any pair of letters, and therefore an Enigma must have an even number of letters. The real Enigma in fact had 26 buttons, and X was used to double up as a space.

The reduced Enigma

The simplest Enigma must have 4 buttons and 4 lights. Figure 2 shows our reduced Enigma.

The wires from the 4 buttons go to a plugboard, which works just like the Enigma's, to mix up the wiring. A wire plugged between any two sockets swaps the original connections.

The real Enigma has 26 sockets, and for practical reasons usually only 6 pairs of plugboard connections were made. There are 100,391,791,500 ways to connect the Enigma plugboard using all six wires.[1]

[1] There are 26 places to put the first plug, then 25 to put the second, and so on. So there are $26\times25\times24\times23\times22\times21\times20\times19\times18\times17\times16\times15=26!/14!$ ways. But for each of the 6 wires, it doesn't matter which way round it is connected: divide by 2^6. Moreover, we can't tell the difference between any of the wires: divide by 6!, the number of permutations of 6 things.

Given there are 4 sockets on the reduced Enigma, so one might think that there are 4!=24 ways of mixing up four wires, but the reduced Enigma's plug board can only provide 10 of these ways! The plug board isn't as good as it is made out to be.

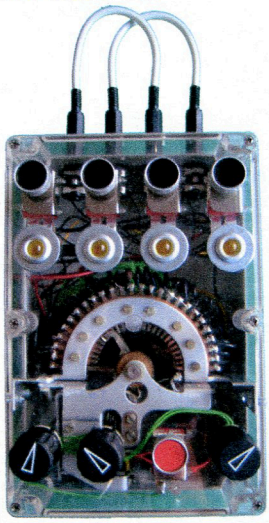

Figure 2. The reduced Enigma in its transparent box. Two wires are shown connected in the plugboard at the top of the machine. The central silver object is the uniselector, which simulates the Enigma's rotors. In this photograph, 25 of its random connections can easily be seen, and the uniselector's 'rotor' is at the sixth position. The knobs and button at the bottom simulate key settings, as explained in the text.

The reduced Enigma is 16×24×12cm, and weighs 2.6Kg — compared to the 12Kg of a real Enigma. Like the Swiss K Enigma, there is a socket for remote lamp connection: messages can be sent across a room and afford some privacy for the sender.

What about the rotors? There are in principle 4!=24 rotors. But most of the rotors don't achieve anything useful. For example, the trivial wiring that takes wire 1 to 1, 2 to 2, 3 to 3 and 4 to 4, does nothing different as it is rotated. Indeed, taking 1 to 2, 2 to 4, 3 to 1 and 4 to 3, and many other possible rotors do nothing as they are rotated. They are therefore quite useless. There are then two rotors (e.g., swap 1 and 2, swap 3 and 4) that have only two different positions. This leaves only 4 rotors out of the 24 that do anything useful!

A reduced Enigma with one rotor (carefully chosen from the 4 non-degenerate rotors) would have a key length of 4. One with two rotors should have a key length of 16. Some combinations of rotors give shorter key lengths.

Since rotors are hard to make, a uniselector was used in the reduced Enigma. (Uniselectors were standard parts of early automatic telephone exchanges, and were also used at Bletchley Park.) The uniselector makes a very satisfying clunky sound as the reduced Enigma is used. The uniselector gives a key length of 50, and it can be wired randomly over the whole length of the key (which rotors cannot be). Each position of the uniselector selects one of the possible substitutions: in principle there are 3^{50} different keys available.

A real Enigma has swappable rotors (to change the wiring) and the rotors rotate. With three rotors, there are 26×26×26=15,576 rotor positions, and they can be put in any order. Later in the war, the rotors were selected from a fixed set of eight, further increasing the number of permutations. The reduced Enigma has the rather more modest arrangement of two fixed rotors — the uniselector is wired to simulate the correct wiring of two rotors. The knobs (bottom left of Figure 2) set the initial rotor positions; pressing the red button resets the reduced Enigma to any of the possible 16 key settings.

The switch (shown bottom right of Figure 2) selects four choices of key length: both rotors locked (a key length of 1, which is useful for demonstration purposes), one rotor locked (a key length of 4), both rotate (a key length of 16), or — for fun rather than accurate simulation! — all

50 positions of the uniselector can be used, giving a maximum key length of 50. (The random wiring of the 34 extra positions was chosen to nearly balance the frequencies of each code.)

A uniselector can easily out-perform the number of permutations of a simple 4-way rotor (and it need have no degenerate connections). A contemporary uniselector could easily out-perform a 26 way original Enigma rotor. But small rotors were more practical for actual use, and they would have been easier to wire up. A single rotor has only 26 internal wires; a single 50 way uniselector has, even for a reduced 4 button machine, 51 wires. Building rotors and swapping them around in the field (as keys are changed) mean that a 26 way rotor is a lot more convenient than a 50 way uniselector.

Conclusions

A real Enigma is a fascinating gadget, but is complex and too valuable to use freely in lectures. Building a reduced Enigma — the simplest possible coding machine that faithfully works like a real Enigma — is a very successful lecturing aid, which allows hands-on use. Its much simplified design allows the key (!) principles of the Enigma code to be explained much more easily. (Being able to lock the rotors is also very helpful.)

More interestingly, the design of the reduced Enigma put some of the design flaws of the real Enigma in a new light. In today's computerised binary world, it is *obvious* that a code machine that *cannot* work in binary has problems. Had Scherbius built a reduced Enigma prototype, perhaps he would have noticed the fundamental problems. Instead, as time went by the Enigma concept was made increasingly complex — adding more rotors, adding a plugboard — without addressing its basic limitations. The basic limitations are very obvious in the reduced Enigma.

Further reading

Simon Singh, *The Code Book*, Fourth Estate, 2000.
David Kahn, *The Codebreakers*, Scribner, 1996a.
David Kahn, *Seizing The Enigma*, Arrow, 1996b.
Friedrich Bauer, *Decrypted Secrets*, Springer, 1997.

Appendix 1: Programming Exercises

The reduced Enigma project required the solution of several problems, which in themselves generate interesting programming problems:

- Which four different letters generate the most words?
  (I chose AEST, though AERT does a bit better but doesn't make such good words.)
- Which rotors should be used, and what is the correct uniselector wiring to simulate the chosen two rotating rotors?
- What is the best (most secure) random sequence that should be used on the 34 uniselector positions that are not simulating the two rotors?
- Finally, how would one build a Bombe simulator to attack the reduced Enigma?

Appendix 2: Key tables and wiring diagrams

Breaking into the original Enigma codes was greatly helped by knowing the wiring and the daily key settings. The reduced Enigma's wiring and key tables (sorry, for January only!) are available from the author: h.thimbleby@ucl.ac.uk