601.445/601.645: Practical Cryptographic Systems January 30, 2020

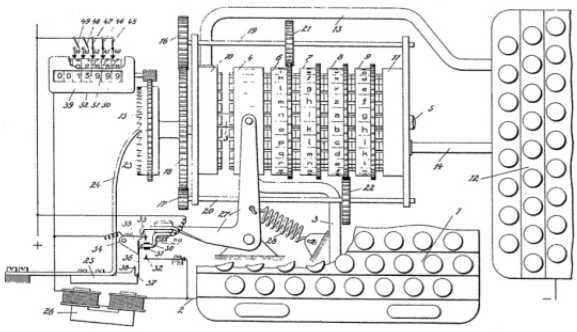
Assignment 1

Instructor: Matthew Green *Due: 11:59pm, February 13*

Name: _____

The assignment should be completed individually. You are permitted to use the Internet and any printed references.

Please submit the completed assignment via Blackboard.



The Enigma machine was widely used by the German Army and Navy during WWII. The design of the machine is now well known, and details can be found in various locations online.[1]

There are a variety of different methods that can be used to break Enigma. Most of those utilized by the Polish and the British at Bletchley Park involved taking advantages of mistakes made by operators or prior knowledge: these included encrypting a sequence of three letters twice at the beginning of a message, not providing random keywords, and not switching out keys in use often enough. The Allies were also able to make sufficient progress through the process of cribbing, which usually involved known plaintext-ciphertext pairs.

For this assignment, you will instead be using statistical methods to conduct an attack on a message that has been encrypted using an M4 Enigma machine.

Problem 1: A statistical attack

You have recovered the ciphertext found in the file **ct.txt**.[2]

[1] See *e.g.,* `https://hackaday.com/2017/08/22/the-enigma-enigma-how-the-enigma-machine-worked/`.

[2] You can find this in the directory `https://github.com/matthewdgreen/practicalcrypto/tree/master/Assignments/Assignment1`.

By chance, you happen to have very concrete suspicions about who the sender is and — perhaps implausibly — you also have access to the exact machine they used! Unfortunately the room the machine was housed in looks like it was recently ransacked. The plugboard is completely unattached, with exactly ten patch cables lying in the floor, and only two of the four rotors remain in place. From this, you are able to uncover the following settings:

Rotor Setup: (reflector) ??? ??? IV III (keyboard)
Initial Start Positions: ??? ??? B Q
Ringstellung: 1 1 1 16
Plugboard: ???
Reflector: C-Thin

You also happen to know that the plaintext is in English, as well as the encoding method used by the sender: before encryption, all spaces are stripped from the message, and any punctuation characters are replaced with the letter X. Using Hill Climbing techniques, come up with your best guess for the message's underlying plaintext. Provide the recovered plaintext, the missing M4 settings, and a Go package named **hillclimb** which conducts the attack as part of your submission.

Your program should have the following command line interface:

```
hillclimb <ciphertext filename>
```

Your program should output (each bullet point represents a single line, separated by newlines):

1. **Rotor order.** A list of four rotor identifiers, beginning with the rotor closest to the reflector. These should be separated by space characters (rotors are identified as `"Beta", "Gamma", "I", "II", "III", "IV", "V", "VI"`).
2. **Start position.** Starting position of each rotor (a letter `A-Z`), separated by spaces.
3. **Plugboard.** A set of 10 plugboard settings, each of which is a pair of letters in lexicographic order, *e.g.,* `BG`, *not* `GB`.

So, for example, output should appear as:

```
I VI IV III
H A B Q
AB CD EF GH IJ KL MN OP QR ST
```

Note: Your code should work on ciphertexts with the same given settings, other than **ct.txt** (we will test this.) There are many Enigma simulators that exist online. You are free to use these for encrypting and decrypting enigma messages. For the purposes of this private assignment submission, you may also repurpose portions of the Go `enigma` (`emedvedev` simulator) inside of your submitted assignment, if this is helpful to you. Please clearly cite this in your code.[3]

[3]You should *not* repurpose or use any other Enigma code in your assignment, nor should you publicly claim credit for another user's code without acknowledgement.

Written Problems

1. Using a standard M3 enigma machine from 1938, how many possible start configurations are there for encrypting a message? Explain explicitly what contributes to the number of possible options and which component contributes the most to this value.

2. Imagine you are working for the Polish Cipher Bureau before 1938 and you manage to uncover the first three letters of a message encrypted using enigma. What information does this give you about Enigma's corresponding settings?

3. Cryptography, and security in general, is a field where the past seems to repeat itself. Do some research into how the Allies broke ciphertexts encrypted using Enigma machines. Choose one specific kind of analysis done by the Allies and describe it in detail. When was it used? Did the Germans counter the attack and if so how? Lastly, at a high level, can you find an analog to this form of attack in modern encryption protocols? Why or why not?

Refereneces

Go

1. Overall introduction to language: `https://tour.golang.org/welcome/1`

2. Installation instructions: `https://golang.org/doc/install`

Hill-Climbing

1. Technical Description of hill climbing: `https://cryptocellar.org/bgac/HillClimbEnigma.pdf`

2. Log N-Gram Probability: `http://alexbarter.com/statistics/n-gram-log-probability/`

3. English monogram, bigram, and trigram frequency sheets: `http://practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/english-letter-frequencies/`

4. Working on breaking enigma on modern cpus: `https://deathofagremmie.com/2012/07/22/completing-the-enigma-challenge/`

5. Attacking short Enigma Messages: `https://web.archive.org/web/20060720040135/http://members.fortunecity.com/jpeschel/gillog1.htm`

Enigma

1. Enigma simulator: `https://github.com/emedvedev/enigma`

2. Cryptanalysis of Enigma: `https://en.wikipedia.org/wiki/Cryptanalysis_of_the_Enigma`

3. Polish Mathematicians work on breaking Enigma: `https://www.ams.org/publicoutreach/feature-column/fcarc-enigma`