The Enigma Machine

by: Alexis Bernazzani and Christopher Vitale

Our Emulated Enigma Machine

Today we're going to demonstrate an emulation of the Enigma Machine, which we created using Python's Integrated Development and Learning Environment (IDLE).

We chose to demonstrate the Enigma machine because not only of its historical significance to the field of cryptography, but we also found it practical to virtualize despite its complex mechanical functionality.

Our goal was to successfully create a virtual Enigma Machine and demonstrate it here at bootCon!

What is the Enigma Machine?

The Enigma Machine was created by the German Government during WWII to provide secure communications between military and political units.

- Required an operator to configure the machine to use a code of the day or time by setting the rotary dials, choosing which rotors to use (and their order), and which letters would be connected to each other on the plug board.

- Would covert the plaintext one letter at a time to ciphertext, which the receiving operator would reverse the process by having the same configurations.

Cryptographic Concepts Applied

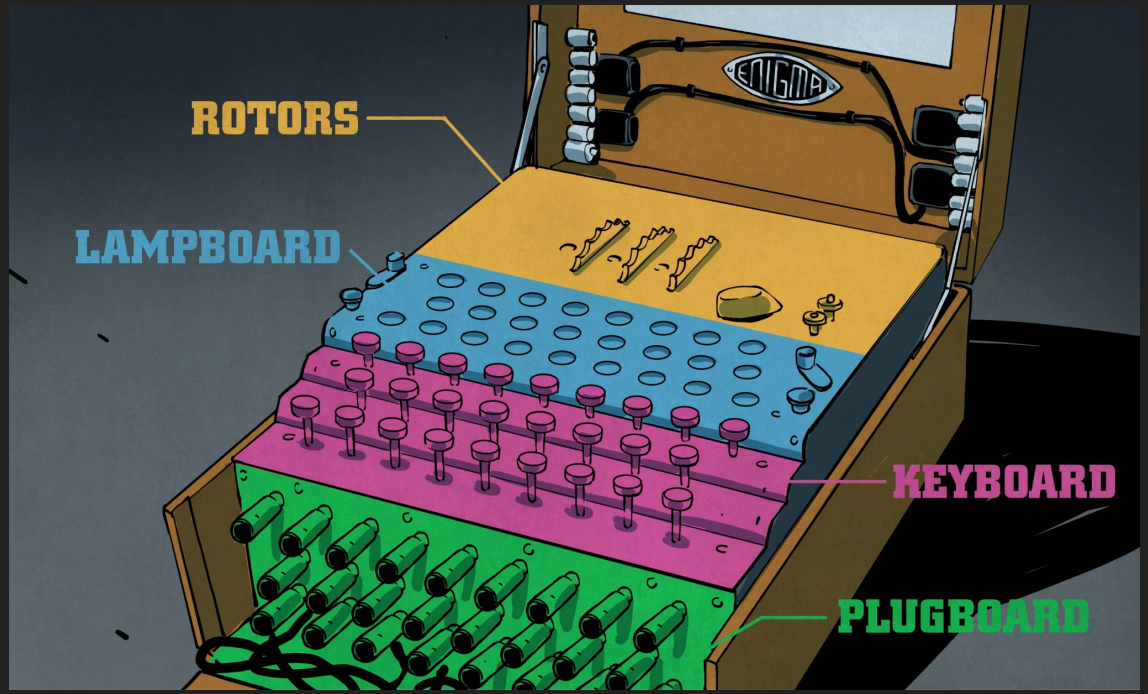
- Ciphertext, Plaintext, Cipher, and Ciphering
- Substitution Cipher
- Polyalphabetic Substitution
- Encryption, and Decryption
- Confidentiality, Integrity, Authentication, and Non-repudiation
- Cryptanalysis

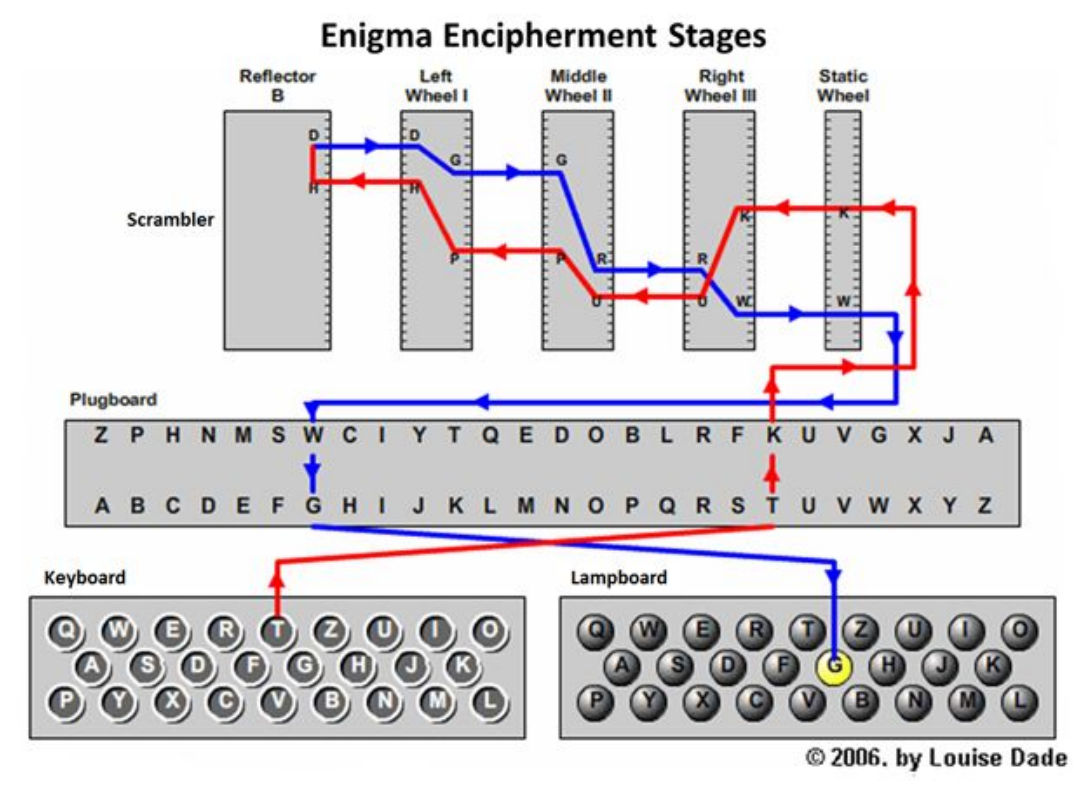
Research steps taken

- Researched the web on the historical background Enigma Machine.
- Watched a video that broke down the main mechanics and internal workings of the machine.
- Occasionally looked at python syntax to supplement source code.

Imagine a circuit that includes a bulb, battery, and switch.

Now imagine several circuits with 26 switches and 26 lightbulbs interconnected…

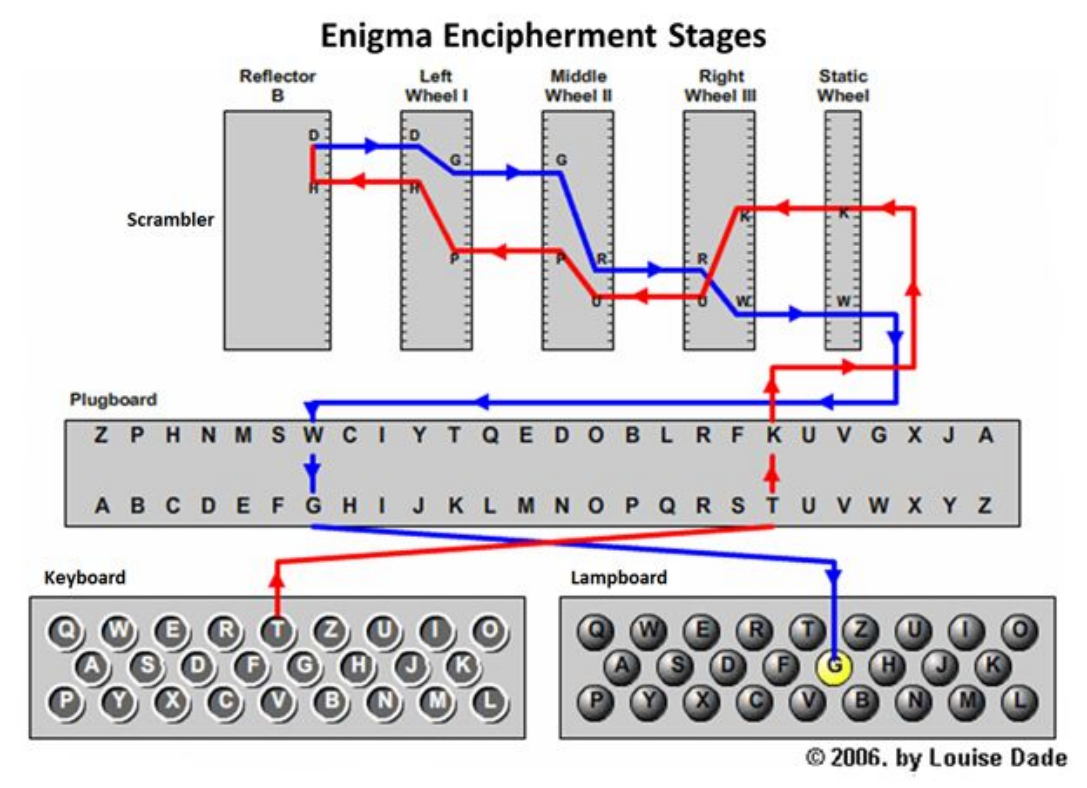




The Order of Current Flow:

- Key Board
- Plug Board
- Static Wheel
- Rotor I (Fast Rotor)
- Rotor II (Medium Rotor)
- Rotor III (Slow Rotor)
- Reflector
- Rotor III (Slow Rotor)
- Rotor II (Medium Rotor)
- Rotor I (Fast Rotor)
- Static Wheel
- Plug Board
- Lamp Board

Figure One shows the plug board on the Enigma Machine. The operator can use up to ten wires to redirect currents going towards the static wheel.



The Order of Current Flow:

- Key Board
- Plug Board
- Static Wheel
- Rotor I (Fast Rotor)
- Rotor II (Medium Rotor)
- Rotor III (Slow Rotor)
- Reflector
- Rotor III (Slow Rotor)
- Rotor II (Medium Rotor)
- Rotor I (Fast Rotor)
- Static Wheel
- Plug Board
- Lamp Board

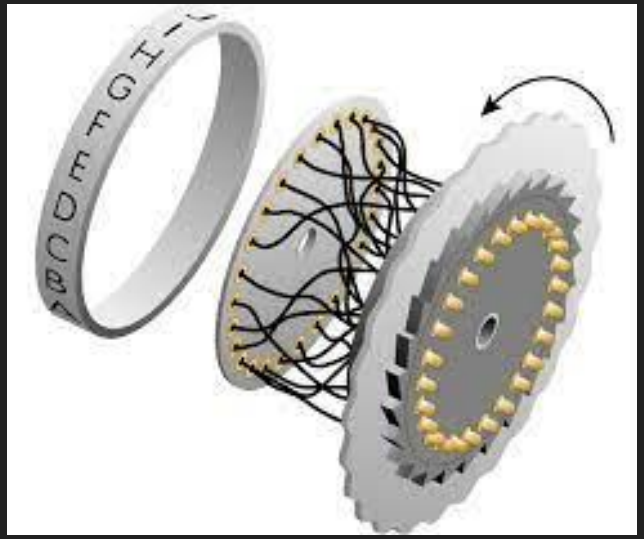
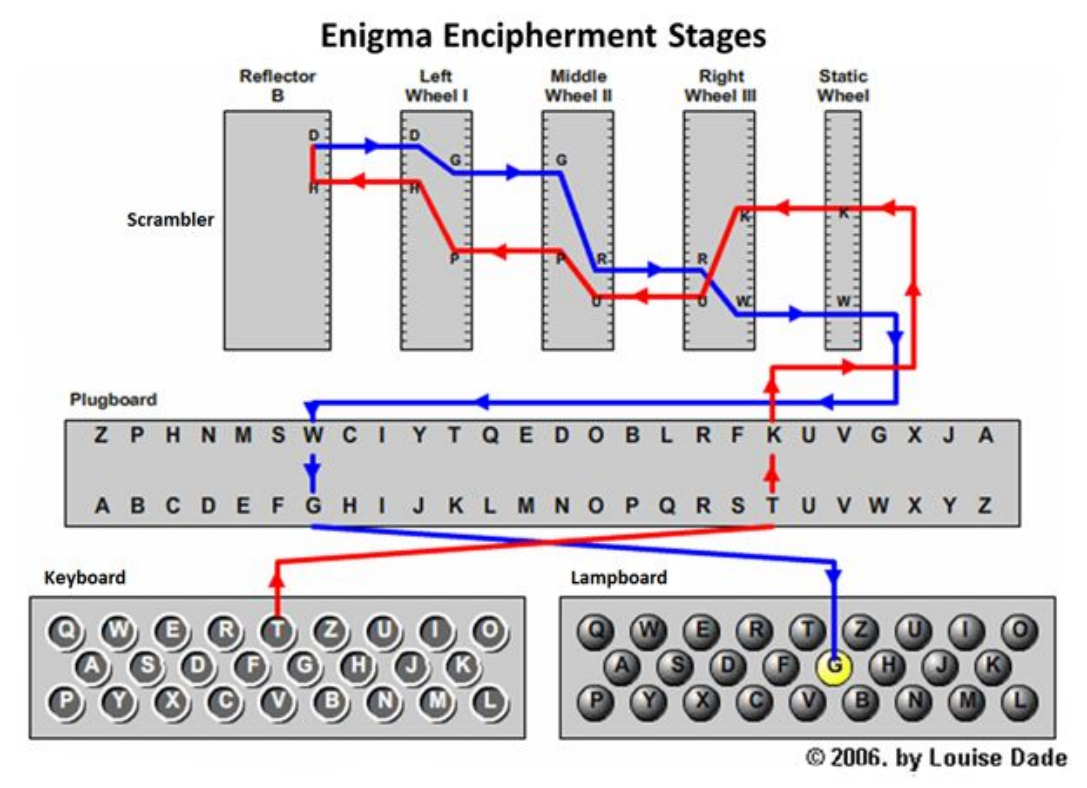


Figure Two shows the internal workings of one of the rotors. Looks the same for all three but with different connections.



The Order of Current Flow:

- Key Board
- Plug Board
- Static Wheel
- Rotor I (Fast Rotor)
- Rotor II (Medium Rotor)
- Rotor III (Slow Rotor)
- Reflector
- Rotor III (Slow Rotor)
- Rotor II (Medium Rotor)
- Rotor I (Fast Rotor)
- Static Wheel
- Plug Board
- Lamp Board

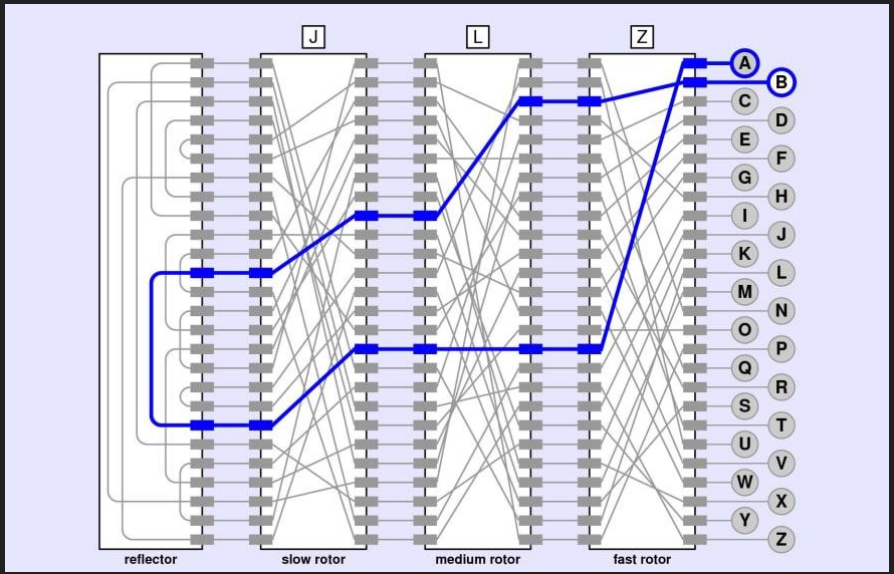
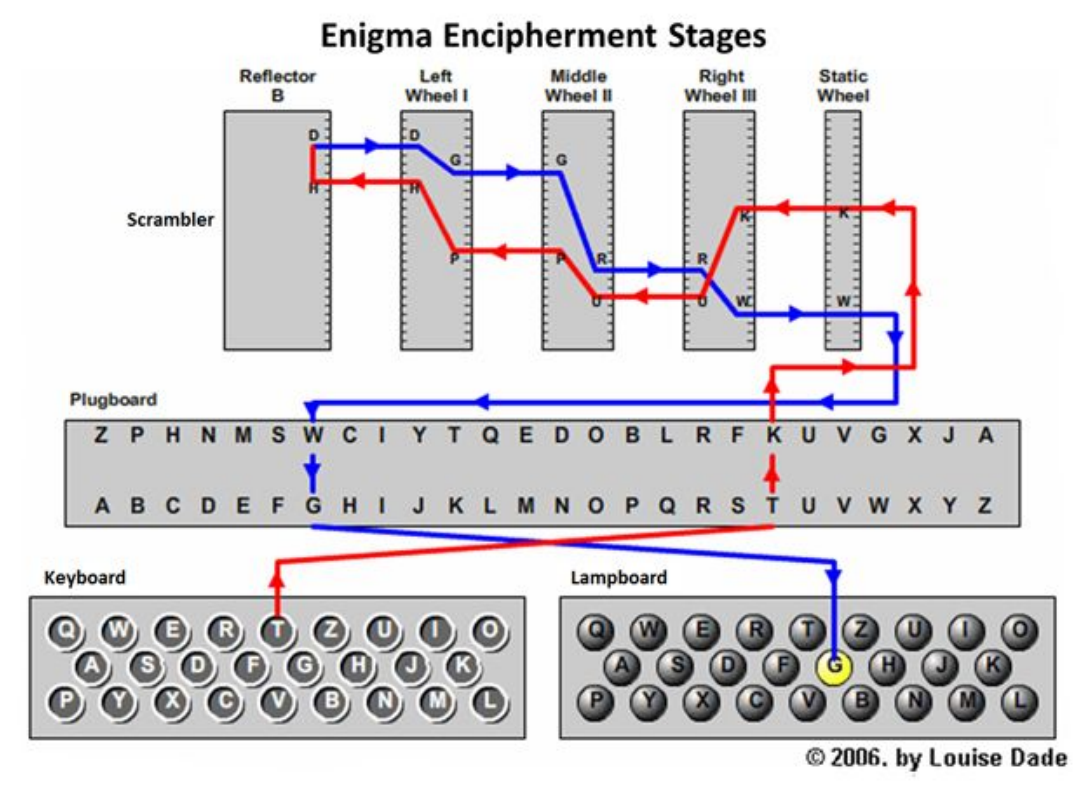


Figure Three shows another example of a closed circuit when a letter is typed on the Enigma Machine (assuming the plugboard isn't being used).



The Order of Current Flow:

- Key Board
- Plug Board
- Static Wheel
- Rotor I (Fast Rotor)
- Rotor II (Medium Rotor)
- Rotor III (Slow Rotor)
- Reflector
- Rotor III (Slow Rotor)
- Rotor II (Medium Rotor)
- Rotor I (Fast Rotor)
- Static Wheel
- Plug Board
- Lamp Board

Enigma Machine Set Up

The Operator can:

- Use up to ten cables on the plugboard to switch letters before the current reaches the static wheel.
- Can choose 3 out of 5 rotors to put into the machine and choose their order.
- Can set the rotary dials to pick how each rotor will start.
- These settings will need to be conveyed to the receiving operator in order to decrypt the ciphertext sent!

The letter pressed on the keyboard can switch up to 9 times before lighting up on the lamp board!

Demonstration Preview

We will briefly go over our source code for the Enigma Machine. A few things to note about our code:

- Our code is written in python.
- The code is broken down into functions, with each function representing a piece of the machine (ex. Each of the three rotors, the plug board, etc.)
- In this demonstration we will briefly cover our code structure and demonstrate how the code emulates an enigma machine.
- We will act as two operators sending and receiving messages with an Enigma Machine.

We will now demonstrate our Enigma Machine created through Python.

Summary of Demonstration

Used lists to define the internal pieces of the machine and thread them through their respective functions.

Lists are mutable objects, which is a key attribute to emulating the complex movement of the rotors. (Taking advantage of the indexes of the list)

The enigma machine was virtualized using python functions to break down the apparatus (such as the rotors, plugboard, reflector, etc.). These functions gave the pieces functionality like the physical machine passing the current through the labyrinthine circuit.

Unlike the actual machine, we allowed input to be words/sentences to make our deliverable more efficient rather than outputting one letter at a time.

As you can see from our demonstration, Alexis configured the Enigma Machine rotor settings that she wanted to use and it produced a ciphertext, then Chris received the configurations from Alexis and the ciphertext to reverse the process to obtain the plaintext.


FIN

Resources

Chapple, Mike, and David Seidl. “Chapter 7: Cryptography and the Public Key Infrastructure.” *CompTIA Security+ Study Guide*, Eighth ed., John Wiley & Sons, Inc, Indianapolis, Indiana, 2021, pp. 179–228.

Owen, Jared, director. *How Did the Enigma Machine Work? YouTube*, YouTube, 11 Dec. 2021, https://www.youtube.com/watch?v=ybkkiGtJmkM&t=540s.

W3Schools. *Python Tutorial*, https://www.w3schools.com/python/.