3 Historical Mechanical Methods (Poly Alphabetic Ciphers)

At this point you should have a good understanding of how the manual substitution and transposition ciphers function by either changing plain text characters for other characters, or by scrambling the plain text; and the methods for attacking or cracking the manual ciphers.

In this section you will continue to learn about the continued back and forth battles between cryptographers and cryptanalysts, where each time the cryptographers come up with a new way to protect messages the cryptanalysts react by finding new methods of attack. We'll continue following the historical timeline, looking at each major step in the evolution because each new development is a building block for modern cryptography. Learning about the developments one at a time makes them easier to digest. And once you understand the building blocks, putting them together to build the modern ciphers will be easier than trying to learn everything at the same time.

The timeline, and the new methods you will learn about look something like this:

1. Cryptographers – develop a new method called Vigenère cipher, or poly-alphabet cipher, for neutralizing frequency attacks.

2. Cryptanalysts – specifically Charles Babbage, develop ingenious method for attacking the Vigenère cipher if the key repeats.

3. Cryptographers – refine poly-alphabetic ciphers to use extremely long keys and develop machines for encrypting and decrypting.

You'll see how time consuming it is to encrypt or decrypt messages using a poly-alphabetic cipher, or a cipher where the alphabet keeps changing, but how poly-alphabetic ciphers can provide very strong security. This should help you understand the situation that led inventors to come up with ways to use machines to encrypt and decrypt messages and speed up the process.

The specific things you should be able to do at the end of this section are:

1. Describe how the Vigenère or poly-alphabetic cipher works. In particular, you should be able to describe how a key word or key phrase is used.
2. Use the Vigenère cipher to encrypt and decrypt messages.
3. Describe how the Vigenère cipher can be attacked
4. Describe the importance of key word or key phrase length, and how key length relates to frequency analysis attacks on Vigenère ciphers

5. List the information that must be shared to encrypt and decrypt messages using a poly alphabetic cipher.
6. Describe how systems that use One Time Pads can achieve perfect security, but the new problem they introduce.
7. Describe in general terms how Enigma and other cryptographic machines function

Required Reading

In addition to reading this document you should read or view the following:

1. Read Singh Chapters 2 - 4

2. Go to the Khan Academy Journey Into Cryptography Website –
   https://www.khanacademy.org/computing/computer-science/cryptography
   View the following videos under the Ancient cryptography section:
   Polyalphabetic cipher
   Polyalphabetic Exploration
   The one-time pad
   Perfect Secrecy Exploration
   The Enigma encryption machine
   Perfect secrecy (One Time Pads)

Optional Reading and Viewing
There are hundreds of web sites and web pages with information on cryptography. Here are a few of the more interesting and more useful sites that I've found over the years.

https://www.commonlounge.com/discussion/b27ffb470c3a4d2b83378edee0a55dab - More information on the Vigenère cipher

https://www.simonsingh.net/The_Black_Chamber/Vigenère_square_tool.html

https://www.simonsingh.net/The_Black_Chamber/vigenere_cracking_tool.html
https://www.cryptomuseum.com/crypto/index.htm
https://pages.mtu.edu/~shene/NSF-4/Tutorial/VIG/Vig-Kasiski.html

Information about Friedman and the Incidence of Coincidence
https://www.nku.edu/~christensen/1402%20Friedman%20test%202.pdf
https://crypto.stackexchange.com/questions/2249/how-does-one-attack-a-two-time-pad-i-e-one-time-pad-with-key-reuse

https://www.tapatalk.com/groups/crypto/the-index-of-coincidence-the-chi-test-the-kappa-t238.html

https://www.cryptomuseum.com/crypto/index.htm - Information about many of the different cryptography machines used in the early 1900s.

Information about the Enigma Machine
https://www.sothebys.com/en/articles/breaking-the-code-the-secrets-of-enigma-cipher-machines
- Watch the video at the end of the page to see an Enigma machine being setup and operated.

https://www.codesandciphers.org.uk/enigma/rotorspec.htm - the actual substitution ciphers on the Enigma rings.

https://www.youtube.com/watch?v=G2_Q9FoD-oQ 158,962,555,217,826,360,000 (Enigma Machine) – Numberphile

https://www.youtube.com/watch?v=V4V2bpZlqx8 Flaw in the Enigma Code - Numberphile

https://www.dcode.fr/trithemius-cipher - Website for enciphering and deciphering with the Trithemius cipher

https://www.dcode.fr/vigenere-cipher - Website for enciphering and deciphering with the Vigenère cipher

Section Content

Introduction - What Is A Poly Alphabetic Cipher?

The advent of cryptanalysis and techniques like frequency analysis brought new challenges to cryptographers and the people who relied on them to protect their secret messages. The kings and queens, and their generals and spies that wanted to keep messages secret came to realize that their substitution and transposition ciphers were no longer sufficient.

One of the new methods that cryptographers came up with a cipher that used a shifting or changing cipher. Instead of always mapping plain text characters to the same cipher text characters or cipher alphabet, these new ciphers would change the cipher alphabet after encrypting each letter. This type of Cipher is called a poly alphabetic cipher because it uses multiple cipher alphabets. This doesn’t mean different alphabets in the sense that different character sets are used, like doing something like changing between English characters and Cyrillic characters. Poly alphabetic ciphers use the same character set but shift the substitution map after encrypting each character.

Another description of the poly alphabetic cipher, a description that may make it easier to envision, is to think of it as a constantly changing series of rotation ciphers. That is, you start encrypting text with one rotation cipher, and then at some point(s) change and use a different rotation.

Here's a quick example with some illustrations to help you understand what this means, and how the poly alphabetic ciphers function.

Say we want to encrypt the plain text “meet me here Tuesday, near reef street”, and we use the following substitution cipher. It’s a rotation cipher that shifts each plain text character by 2 places.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B

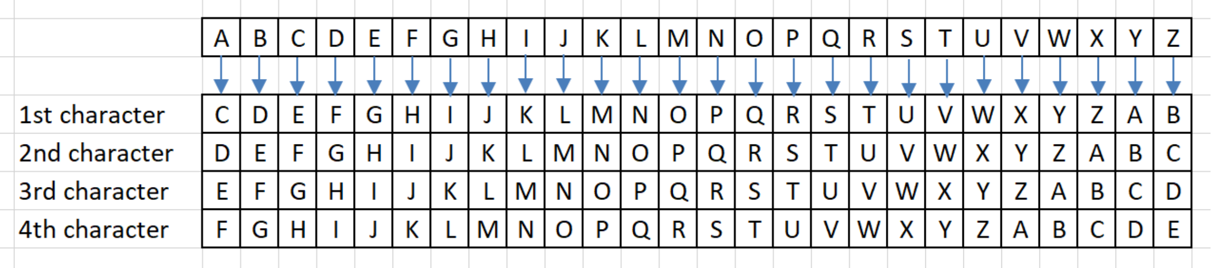
This results in the following cipher text:

```
meet me here Tuesday, near reef street
OGGV OG JGTG VWGUFCA, PGCT TGGH UVTGGV
```

This resulting cipher text is very simple to crack using frequency analysis. Especially since it has so many “e” characters and “t” characters.

This method of encrypting is called a mono alphabet cipher because the cipher alphabet never changes. And never changing is what causes the weakness in the cipher, because the letter **E** always maps to the same letter in cipher alphabet. In this example **E** always maps to **G**, which will be a dead giveaway when we run the frequency analysis.

But a few cryptographers had the great idea of changing the cipher alphabet *during* the encryption process. For example, one of the ideas was to change the cipher alphabet each time another character was encrypted by shifting the characters in the cipher alphabet to the left. The following figure shows how the cipher text would be shifted for the first four characters to be encrypted.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1st character	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
2nd character	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
3rd character	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
4th character	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E

This is called a poly alphabetic cipher because it uses a different rotation, resulting in different cipher alphabets for each substitution. Using a cipher where the rotation changes for each character results in the following cipher text:

```
meet me here Tuesday, near reef street
OHIY SL PNBP FHSHTRQ, GYVN OCDF TVUIJZ
```

As you can see, the first two **E** characters in the word **meet** are encrypted to different characters, **H** for the first **E** and **I** for the second **E**. Mapping the plain text **E** characters to different cipher text characters will cripple any frequency analysis, by flattening the letter distribution in the cipher text. This is much the same result as with homophonic substitution, but unlike homophonic substitution doesn’t require the use of all the extra cipher text characters.

To summarize, a poly alphabetic cipher is one that uses different cipher alphabets during the encryption process. And changing the cipher alphabet doesn't mean using a different character set, it just means somehow changing the order of the characters in the cipher alphabet.

History and Variations of the Poly Alphabetic Cipher

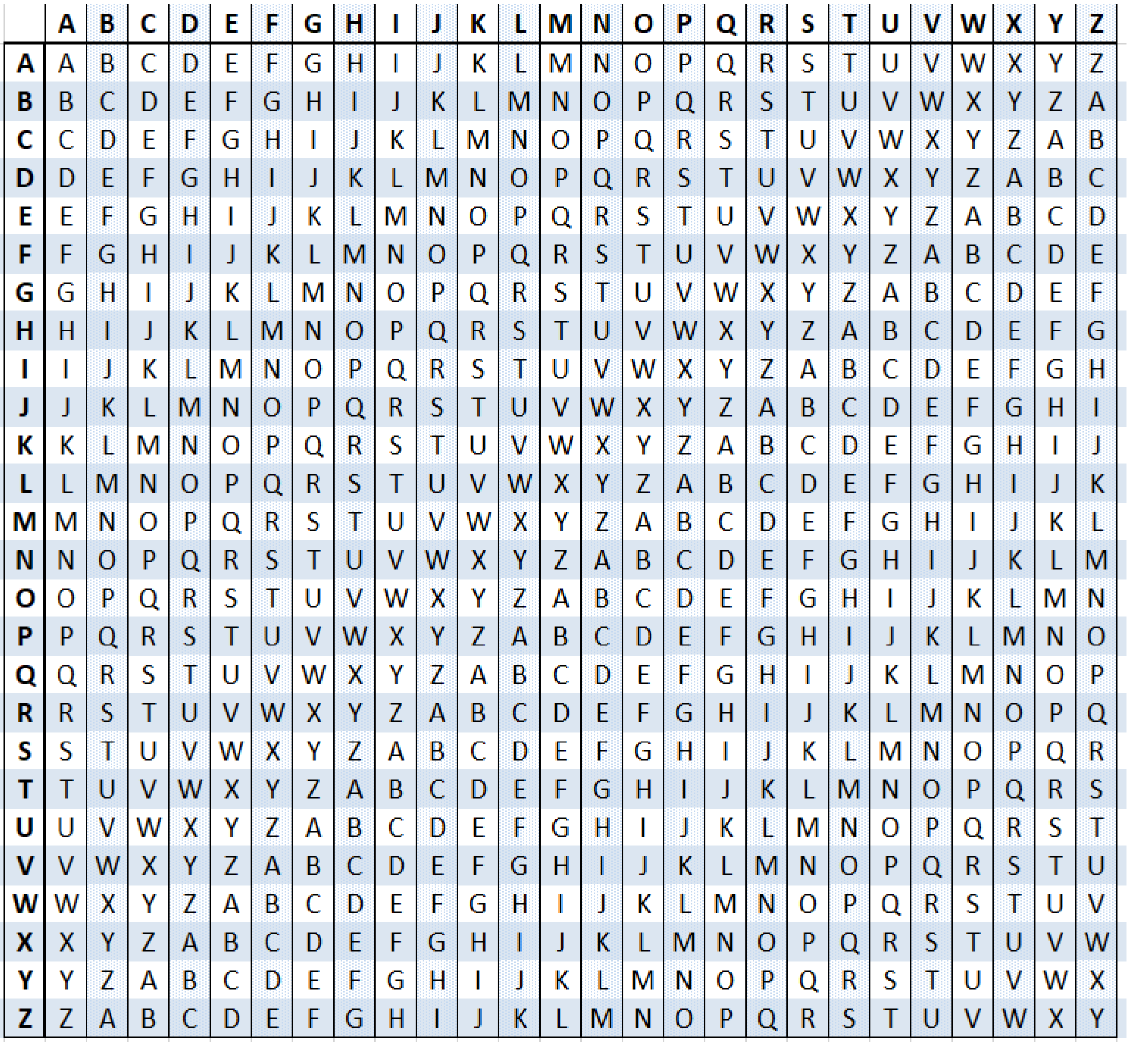
The credit for the invention of poly alphabetic ciphers, like most everything in cryptology, is a little skewed. The first written record is by an Arab scholar named Al-Qalqashandi in the late 1300’s, but there are claims that it was actually developed by the Arab cryptologist named Al-Kindi nearly 500 years earlier. This is the same Al-Kindi who first wrote about using frequency analysis to crack messages.

Alberti Cipher

In Europe the first documented use of a poly alphabetic cipher was credited to an Italian named Leon Battista Alberti. Alberti developed an encryption scheme where he would start with one cipher alphabet and then periodically change alphabets. His scheme wouldn't change alphabets at regular intervals, he would just change them when he felt like it. To allow the message to be decrypted he would indicate that the alphabet had changed by including an uppercase letter or a number in the cipher text. While Alberti's scheme was an improvement, cryptanalysts were able to pick up the fact that the uppercase letters and numbers signaled a change in alphabets.

Trithemius Cipher

The next improvement was by a German Monk named Johannes Trithemius who developed a cipher where each character was encrypted using a different alphabet, in a way very similar to the one we used in the previous example. That is, the alphabet changed with every character not at random intervals like with the Alberti Cipher. The Trithemius Cipher used something called a tabula recta, which is Latin for right table or straight table, to determine the cipher alphabets and encryption mappings. An example of the tabula recta is shown in the figure below.



	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Encrypting text with the Trithemius cipher required a bit of work. The first plain text character was encrypted using the cipher alphabet in the first row, or the **A** row of the tabula recta. As you can see this won't make any change to the first character but that's okay, the characters will start changing soon.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

The second plain text character is encrypted using the next row down which we also call the **B** row because it starts with the letter **B**.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A

The third plain text character is encrypted using, yes you guessed it, the next row down which we also call the **C** row because it starts with the letter **C**.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B

This continues with each subsequent character using the next row in the table until the last row is encountered, at which point the “A” row is used again.

Here’s an example of encrypt the following phrase. Note that the spaces will be removed.

I AM THE WALRUS THEY ARE THE WALRUS GOO GOO GJOOB

(If you want to follow along it helps to have a copy of the table handy.) The first plain text character is **I**. Using the **A** row to encrypt the letter **I** results in a cipher text character of **I**. The second plain text character is **A** and using the **B** row, or the second row of the table, to encrypt the **A** results in a cipher text character of **B**. The third plain text character is **M** and using the third row of the table, or the **C** row, results in a cipher text character of **O**. The fourth plain text character is **T** and using the fourth row of the table, or the **D** row, results in a cipher text character of **W**. (I’ll stop the example here because I’m sure you understand the process at this point, and don’t want to be tortured by the details of encrypting every character.) The end result is the encrypted text:

IBOWLJCHTAEDFUSNQIWMBZSXJQUTIRSLUVOSYZN

Trithemius Cipher - Technical Aspects

One of the great aspects about the Trithemius cipher is that it doesn’t require exchanging any information between the sender and the recipient. That is, as long as both parties know how to set up and use the table. While setting up the table is fairly easy, the Trithemius Cipher is harder to use than a normal substitution cipher because it requires keeping track of which row or alphabet to use for every character you encrypt or decrypt. At the time it was first invented, using the Trithemius Cipher was worth the extra work because of the extra security it provided by flattening the frequency curve. And since it changed alphabets with every character it didn’t require adding characters to the cipher text to indicate the change like the Alberti cipher, which meant that there was no obvious sign to cryptanalysts that the cipher alphabet was changing.

However, once cryptanalysts caught on to the use of poly alphabets any messages encrypted with the Trithemius Cipher proved easy to break since they used such an obvious and regular process for changing cipher alphabets. That is, all it takes to decipher a message is the table. It’s relatively easy to check enciphered text by working backwards through the table, and you always know that the first character was enciphered with the **A** row, the second character was enciphered with the **B** row etc.

Vigenère cipher

The most well-known historical variant of the poly alphabetic ciphers is called the Vigenère cipher which was documented in the 1500s. Like many other famous ciphers, the Vigenère cipher is actually misattributed and named after the wrong person. It’s named after Frenchman Blaise de Vigenère, but it was actually first written about by Giovan Battista Bellaso, an Italian.

The Vigenère Cipher works very much like the Trithemius Cipher with one critical improvement. It also uses the tabula recta although in this case it's called the Vigenère table. The table is the same but the way you move through the cipher alphabets is the significant difference. Instead of always following the same pattern and encrypting a plain text character and then always moving down one row to encrypt the next character, the Vigenère cipher uses a keyword or key phrase to determine how to move through the rows. For example, if the keyword is **flux capacitor** the first plain text character will be encrypted using the **F** row, the second plain text character will be encrypted using the **L** row, the third plain text character will be encrypted using the **U** row, the fourth plain text character will be encrypted using the **X** row, etc. If the keyword isn’t long enough to encrypt the entire plain text, then the characters in the keyword are repeated.

Here’s an example of using the key phrase **flux capacitor** to encrypt the following phrase. Note that the spaces will be removed.

I AM THE WALRUS THEY ARE THE WALRUS GOO GOO GJOOB

(If you want to follow along it helps to have a copy of the Vigenère table handy.) The first plain text character is **I** and the first character in the key phrase is **F**. So, the **F** row will be used to encrypt the **I** (instead of using the **A** row), which results in a cipher text character of **N**.
The second plain text character is **A** and the second character in the key phrase is **L**. So, the **L** row will be used to encrypt the **A** which results in a cipher text character of **L**. The third plain text character is **M** and the third character in the key phrase is **U**. So, the **M** row will be used to encrypt the **M** which results in a cipher text character of **G**. (You can continue this process if you want, but I’m going to stop the example here.)

The resulting cipher text is:

NLGQGGWPLTCLHYJJUODVHTWCTKIJLZIDNQGYOQJ

Vigenère Cipher - Technical Aspects

The Vigenère cipher is almost as easy to use to encrypt text as the Trithemius cipher. That is, assuming the sender and the recipient know how to set up and use the table, the only additional information they need to exchange is the keyword or key phrase. The actual steps in the encryption and decryption process are also as easy as with the Trithemius cipher. Which is to say they're a little harder than with a substitution cipher that uses a single alphabet, but still simple enough that the process can be done manually.

But even though the Vigenère cipher is almost as easy to set up and use as the Trithemius cipher it is much more secure. Remember that if you suspect that the Trithemius cipher is being used, it's simple to check because the cipher always uses the same pattern to move through the rows in the table. But with the Vigenère cipher there's no way to know which rows to use unless you have the key. This made the Vigenère cipher so strong that it was considered to be unbreakable for almost 300 years earning it the nickname “le chiffre indéchiffrable” which is French for 'the indecipherable cipher'.

The Important Information Regarding Poly Alphabetic Ciphers

Like everything you're learning about in cryptology, you've been presented with a lot of information on poly alphabetic ciphers. There are interesting historical stories, and the names and dates of people who are credited, rightfully or not, of making advances in the ciphers. You don't really need to memorize or remember all the background info, but you should make sure you understand the basics of how the Vigenère cipher functions. The crucial piece is how the cipher shifts based on a key. It's important that you grasp this concept because modern stream ciphers use a similar process. If you understand this piece with the Vigenère cipher and can see or visualize how the key causes the cipher to shift, it will make it much easier to understand modern stream ciphers.

Cracking the Vigenère Cipher

The Vigenère cipher is certainly tough to break. I know there's no way I could think of a way to crack it. And, it's so strong that under certain conditions it's impossible to break. Let me repeat that, the Vigenère cipher is so strong that under certain conditions it will be impossible to break.

There are two main factors that will determine whether cipher text encrypted with the Vigenère cipher can be broken or not. The first is the length of the keyword or key phrase, and the second is whether the keyword is used to encrypt more than one plain text message.

When we talk about the length of the keyword, we're not concerned about its absolute length. Determining the keywords strength can only be done by comparing the keyword's length relative to the length of the plain text. That is, there's no magic number of characters or set length that will make a keyword strong or weak. It's impossible to say that a 1000 character keyword is good, or if it's bad. What makes or breaks the keyword is how long it is compared relative to the length of the plain text.

Historically, when the encryption was being done by hand, the keywords or key phrases were relatively short because this made them much easier to remember. When the keywords are of practical, manageable size the resulting cipher will have a small flaw. This flaw was discovered by two incredibly intelligent researchers at roughly the same time in the mid-1800s. German cryptographer Friedrich Kasiski and Englishman Charles Babbage (yes, that Charles Babbage), independently discovered this flaw and developed a method for using the information it revealed to break the unbreakable cipher.

Flaw 1 - Using Short Keys to Crack Messages

This method is called the Kasiski's method as Kasiski was the first to publish it. But it turns out Babbage may have discovered it earlier but just never got around to publishing it. I'm not going to go into the details of their method as the Singh book has an excellent explanation. And it's not critical that you know exactly how it works, but you should be aware of the following facts:

1. Kasisiki's method relies on the fact that repeated use of the key will result in patterns of repeated text that can be found in the cipher text. These patterns may be subtle, but nonetheless they provide enough of a crack to allow the cipher text to be broken by revealing probable key lengths. Once the key length is discovered the cipher text can be broken into sections that were encrypted using the same alphabet. For example, if you know that the key word is 7 characters long you know that every 7th cipher text character was encrypted using the same alphabet. Performing frequency analysis on each of the groups of characters that were encrypted using the same letter from the keyword will reveal the same clues and information that can be found from the frequency analysis of any mono alphabetic text.

2. These patterns will occur when the keyword or key phrase is reused, which happens any and every time the keyword is shorter than the plain text. For example, if the key contains 6 characters and the plain text is 48 characters then the key will have to be used 8 times. In most cases the plain text messages were much longer than the key.

3. The way to avoid this flaw is to use keys that have as many characters as the plain text. I know this is probably obvious, but it makes coming up with keys much more difficult. For example, it would be easy to come up with and remember a key that had 10 or 20 characters, but much harder to come up with keys with 1000 or 2000 characters. You might think that it wouldn't be hard if you wrote a computer program to pick keys for you. But remember that there were no computers in the 1500s.

Flaw 2 - Using Repeated Keys to Crack Messages

The second flaw occurs when the same key is used with multiple messages. If the same key is used for even two messages, there are methods that can be used to crack the message. Most of the work on this method was done by William Frederick Friedman in the early 1900's. Friedman was an expert in cryptology who provided immense support during World War II, and went on the lead the cryptography division at the NSA when it was formed. His publications on cryptography are still considered as some of the best in the world.

The techniques Friedman developed use a combination of linguistics and statistics, and to be perfectly honest I barely understand how they work. I can follow some of the logic he uses for something called the Index of Coincidence, but I get lost pretty quickly when I read about further attacks called the kappa, chi and phi tests. (These names are from the Greek letters κ for kappa, χ for chi and φ for phi, which all have special meaning in math). Once again, it's not super critical that you understand the details of the attack, but you need to understand these general parameters:

1. It relies on the fact that repeated use of the same key, in encrypting more than one message, will result in patterns that will allow the cipher text to be broken.

2. The way to avoid this flaw is to never use the same key to encrypt more than one single message.

One Time Pad (OTP) - Perfect Secrecy

The information you've been presented about cryptology and ciphers has been coming fast and thick. You've been exposed to so much new information that it's possible that the last critical points about the Vigenère cipher may have not fully registered. Because these features of the Vigenère cipher are key building blocks to modern cryptographic methods it's really important that you appreciate how important they are, understand their impact, and can visualize what how they impact the ability to protect messages. To ensure that you understand these points we're going to emphasize them by going over them one more time and discuss how they can be implemented.

The facts that we're talking about are (1) that the Vigenère cipher can be used to achieve perfect, unbreakable encryption if the keyword is as long as the plain text, and (2) if the keyword is only used to encrypt one message and is never reused. Take some time and let the fact that the Vigenère cipher can be used to achieve perfect, unbreakable secrecy. If it's used with a key that's is as long as the cipher text, then no one will be able to crack the message.

And we're not just talking about manually decrypting the message. Even an attacker with the most powerful modern computer won't be able to crack the message. The problem that an attacker has is that each cipher text character can be any plain text character, so there will be an infinite number of possible solutions, all of them equally likely. For example, if there are even just 5 characters of cipher text, these 5 characters could be any 5 plain text characters. And how many 5 character words can you think of?

> Alpha, radio, chain, donut, worry, grade, value …

I'm only guessing about the total number, but I guess there must be tens of thousands of 5 letter words. And the possibilities only increase when you consider that the spaces between words have been omitted. So, these 5 characters could be a 1 character word followed by a 4 character word, or a 2 character word followed by a 3 character word, etc. You could write a computer program that would show you all the possible words or word combinations, but you'd still be faced with the problem of choosing which one was correct. In other words, solving this problem is a problem that more computing power is not going to help with. A computer could run through all different combinations of 5 letters, throw away nonsense strings like **aabbb** and keep ones that are actual words. But you'll still be faced with reading the resulting list of words and trying to figure out which one is correct.

Any of the 5 character solutions is equally as likely as any other possibility. And with longer messages the number of possibilities only increases. Just think of how many different possible messages could be written with 100 characters, or 1000 characters. Any and all of them could be the correct message. The *only* way to know the correct decryption is if you have the key.

The other important fact is that the key word cannot be reused, it must only be used once. Using a key more than once provides attackers with a slim finger hold that they can leverage and use to eventually break the key.

This means that to achieve perfect secrecy with the Vigenère cipher the sender and the recipient need to agree to use keys that are as the same length as the message, and only use each key one time. Now the trick to achieving perfect secrecy becomes coming up with a system for generating keys, generating enough for keys practical use, exchanging the keys between the sender and recipient, and making sure the sender and recipient coordinate and use the same keys for the same message.

The name for this type of system is the One Time Pad (OTP). It gets this name because it originally consisted of a pad of paper. Each paper in the pad had one key written on it. There are two copies of each pad, one for the sender and one for the recipient. The first message is encrypted with the key on the first sheet of paper. The sheet of paper and its key are then destroyed to ensure they’re only used one time. When the first message is received, it’s also decrypted using the first key. And the recipient then destroys the first key to ensure it isn’t used again.

The big problem with the OTP system is exchanging the one-time pads. If you’re worried about someone stealing your secrets and want to be sure they are delivered securely then you can't send your one-time pads electronically. The one time pads have to be printed and delivered by hand. I’m sure you can envision the problem of delivering pages of codes to different people around the country or around the world. It might seem like something from a bad spy movie, but at one time businesses really did use couriers with a briefcase handcuffed to their wrist so the one-time pads couldn't be stolen.

And the delivery of the one-time pads wasn't a one-and-done occurrence, it had to happen frequently enough to ensure that the recipient and the sender always had fresh keys. The delivery schedule required balancing the convenience of sending a lot of keys, so the exchange doesn't have to happen so frequently, against the risk of having the keys stolen. If the keys are stolen or compromised, then it's better to have a smaller set. But that means having to exchange the keys more often.

And many government agencies and businesses need secure communications with not just one other person, but with many other people or locations. Of course, this requires exchanging additional sets of one-time pads, which further complicates the key exchange process.

This is one more concept that's super important to understand because it still affects cryptography today. That is, the one-time pad can provide perfect security but exchanging the one-time pads and keeping them secure is very difficult. This problem vexed cryptographers until it was finally solved many years later, which you’ll learn about soon.

Rise of the Machines

In the ongoing battle between cryptographers and cryptanalysts, the use of the Vigenère cipher with a One Time Pad swung the advantage so far in the favor of the cryptographers that you might think the battle is over and the cryptographers have won. After all the use of the OTP provides perfect security, so there's just no way for messages encrypted using the OTP to be broken. Of course life can't be that simple, and it turns out that even though One Time Pad encryption method is perfectly secure, generating and exchanging the One Time Pads introduced new flaws that cryptanalysts were able to take advantage of, and once again the battle continued.

As you just learned the One Time Pads needed to be long, or have a lot of random characters, and they needed to somehow be exchanged in a secure manner. In the early 1900's several ingenious inventors and cryptographers came up with ideas for mechanizing or automating parts of the encryption and decryption processes. They also used some rudimentary electronics, but this was before the advent of computers, so the electronics were very crude and simple by today's standards. The basis of these machines was something like the old decoder ring. But instead of setting the decoder ring and using the same rotation for every plain text character, these machines would mechanically rotate the rings to change the rotation for each plain text character. And, the machines would vary the rotation to act much like the one-time pad. That is, they wouldn't act just like a straight Vigenère table and shift the cipher alphabet by one character, the shift followed a scheme, but it wasn't a simple scheme.

There were hundreds of different variations of these machines developed by different nations and organizations. Some of the most famous of these machines include the Enigma systems used by the Germans in World War II.

These machines were able to use mechanical parts to help generate all the characters needed by a one-time pad. If the sender and recipient each had a copy of the machine, they could exchange messages without having to go through the process of constantly generating and exchanging one time pads. But the machines didn't totally solve the problem of exchanging secrets because the sets of characters they would generate for the one-time pad would eventually repeat, the set they would generate would be very long, but it wasn't infinite.

And the machines also introduced one new problem. In most versions of the machines, both the sending and receiving machine were customizable. This was done to prevent the second problem of OTPs, where the system could be broken if the same key was used more than once. For example, the German Enigma system had multiple sets of rotors that needed to be installed in a specific sequence to initialize the system. One version of Enigma had 5 rotors, of which 3 were used at any time. And each rotor could be initially set in 26 different positions, plus there were 10 wires that needed to be set to one of 26 positions at either end. This resulted in ~1.5x10^{20} possible initial settings. The setup wasn't difficult, but it resulted in an insurmountable number of possibilities for anyone trying to guess the initial setup. But it also meant that typically each day all the machines in the system needed to be reconfigured to use the same initial settings.

In essence, the cryptographic machines just traded one problem for another. They no longer needed to exchange the OTPs, but they needed to exchange the configuration settings. The configuration settings were much simpler and shorter than the stream of characters in an OTP.

But they still had to be written down and exchanged, creating a vulnerability in the overall system.

How Enigma and Other Mechanical Encryption Machines Work

The Enigma encryption machines, and other mechanical encryption devices were able to provide an encryption scheme that was excruciatingly difficult to break. The machines provided a mechanical way to simulate a poly alphabet cipher, like the Vigenère, and provided nearly unbreakable encryption.

In this section you'll learn some details about how the Enigma machines are constructed and how they work using a simple example. After that, you'll learn how the results from the simple example can be extended to the full Enigma machine. This information isn't crucial to the big picture of how modern ciphers work, but it is an interesting side story.

Design and Construction

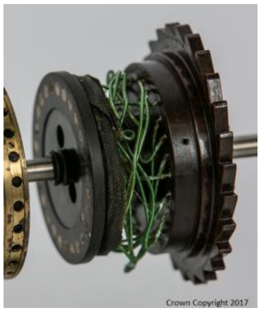
The design of the Enigma machines was deceptively simple. It consisted with a set of keys for entering the data, which looked like an old typewriter, and a set of corresponding lights which would show the encrypted characters. The device was powered by a battery, and when a key was pressed it would complete an electric circuit, acting like a light switch. The electric current from the battery ran through a set of wires inside a set of rotors, creating a circuit which ended at one the light bulbs.

The Enigma Machine[i]

The actual encryption was performed by a set of rotors, which would rotate each time after a character was encrypted. The original Enigma machines had 3 rotors, while later models added a fourth. The rotors could be easily removed from the machine and replaced in a different order, which was done at a certain time every day.

Enigma Rotors[ii]

Each rotor was hollow with metal pins, or contacts, on both sides. Inside the rotor was a set of wires which connected the pins from the left side of the rotor to the pins on the right side. Each rotor has 26 pins each side, one for each letter of the alphabet. These wires did not go straight across, instead they connected pins in different positions on each side of the rotor, which built an electric version of a substitution cipher table. That is, they might map the **A** pin on the left side of the rotor to the **J** pin the right side. Each of the 3 (or 4) individual rotors had a different wiring scheme, which in essence created three different substitution tables.

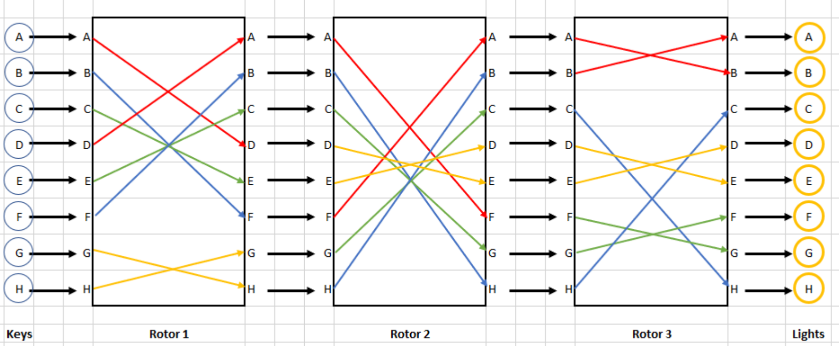


Crown Copyright 2017

Rotor Exploded View[iii]

A Simple Example

The following diagram illustrates how the wiring inside the rotors builds a circuit between the pins on either side of the rotor, and how the pins on one rotor continue the circuit by connecting to the pins on the adjacent rotor. For clarity' sake it's been simplified to only use 8 wires to map 8 letters, instead of the full 26 which would be difficult to follow. And even though we're only using 8 wires, the concepts will still be the same, and can easily be scaled up to 26.

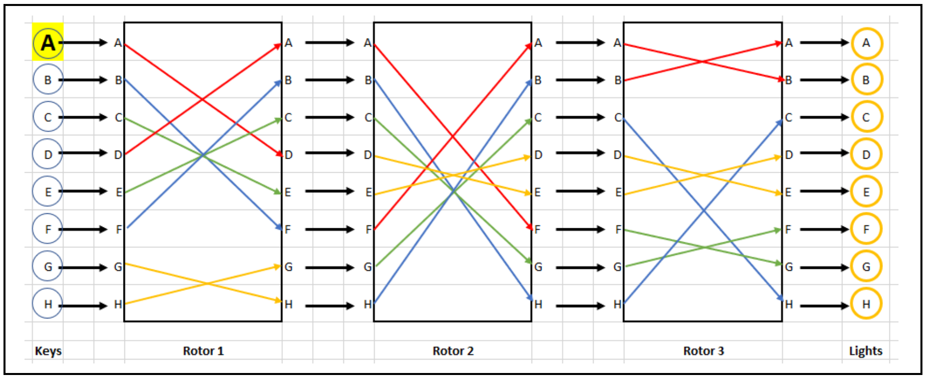


Simplified Rotor Wiring

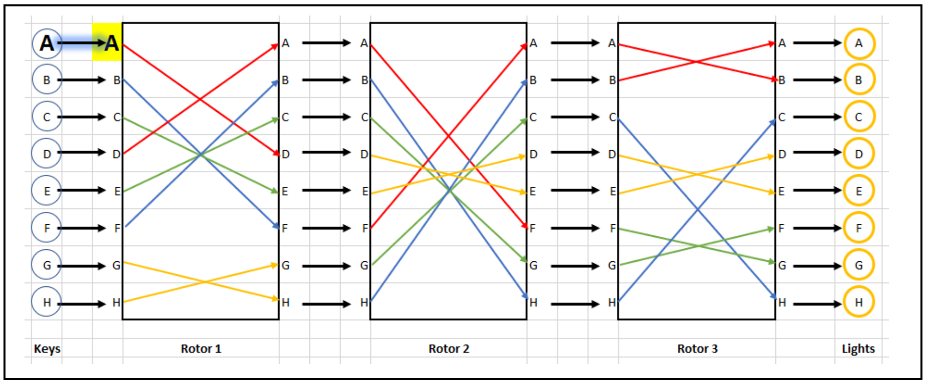
When the operator pressed a key to enter a character, an electric signal would run into a pin on the left side of the first rotor. The wire inside the rotor would then move the signal to a different pin on the right side of the rotor. The pins on the right side of the rotor would be touching a pin on the second rotor, allowing the signal to enter the second rotor. The wire inside the second rotor would then move the signal to a different pin on the right side of the rotor. The process was repeated in the third rotor, where the signal moved from the pin on the left, through the wire to a different pin on the right. At this point the circuit would be complete and one of the lights on the lamp board would light up.

The diagrams on the following page illustrate the process of building the circuit, from pressing the key to moving through the rotors on the selected wires, and finally lighting a light.

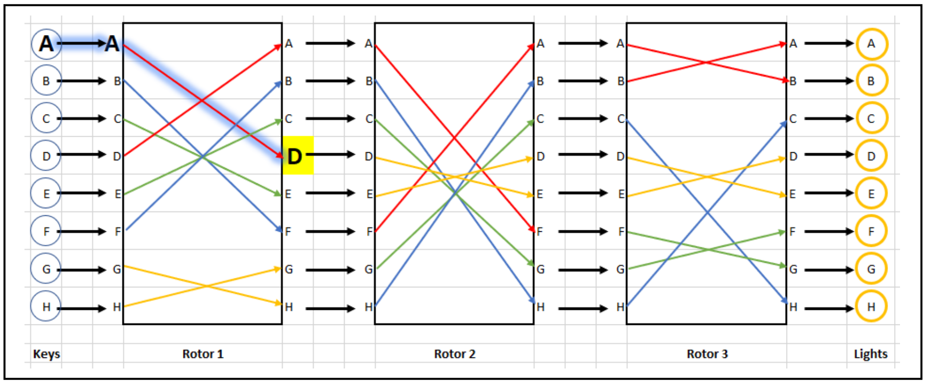
Step 1 – The operator presses the **A** key on the keyboard.



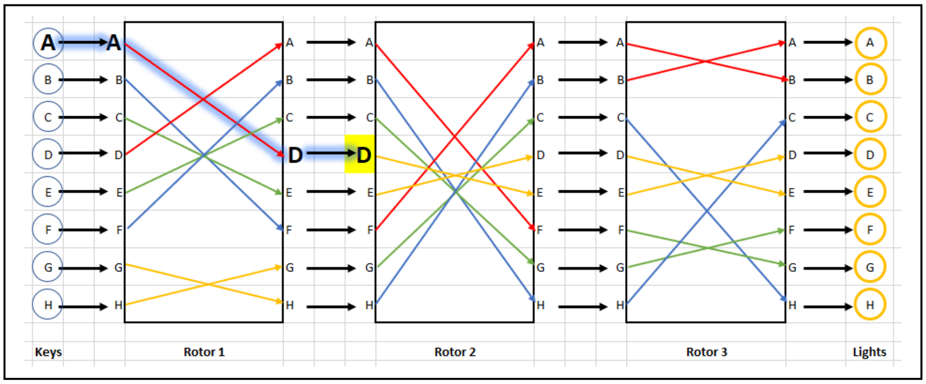
Step 2 – The key press connects an electric current from the battery to the **A** pin on the left side of the first rotor.



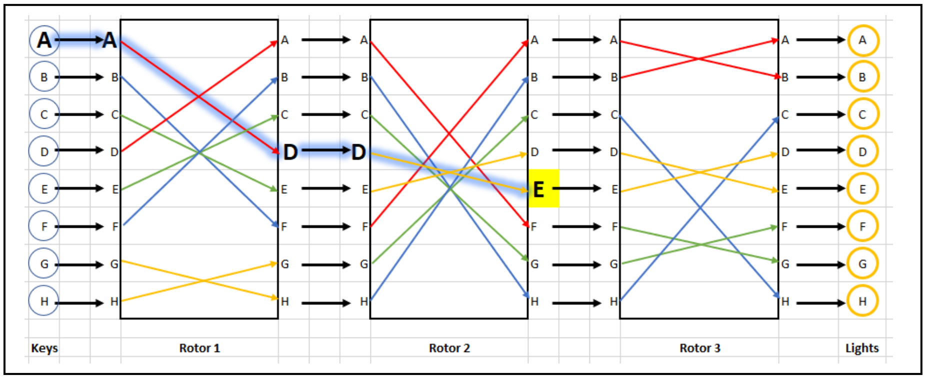
Step 3 – Inside the rotor, the wire connected to the left **A** pin moves the electric current to the **D** pin on the right side of the first rotor.



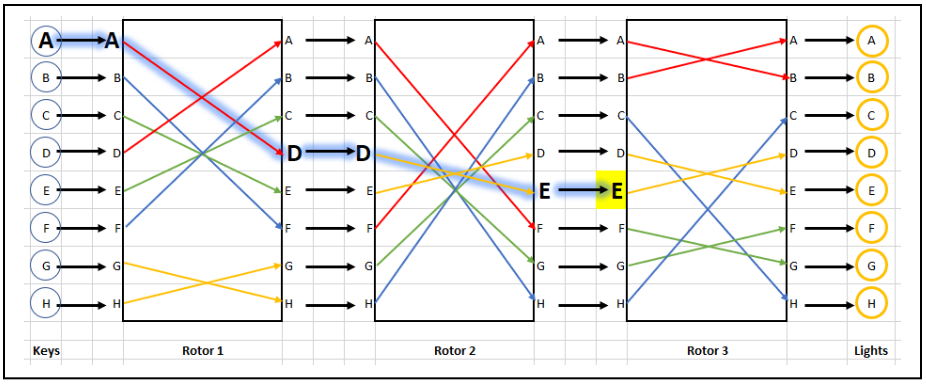
Step 4 – The **D** pin on the right side of the first rotor is in contact with the **D** pin on the left side of the second rotor. This allows the current to move to the second rotor.



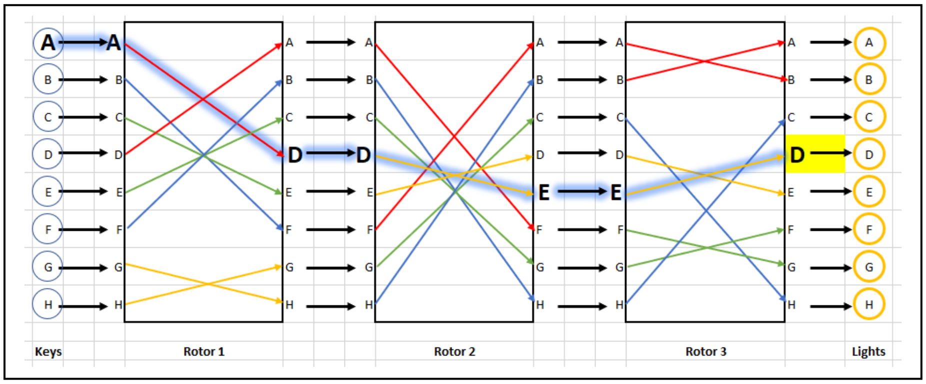
Step 5 – Inside the second rotor, the wire connected to the left **D** pin moves the electric current to the **E** pin on the right side of the rotor.



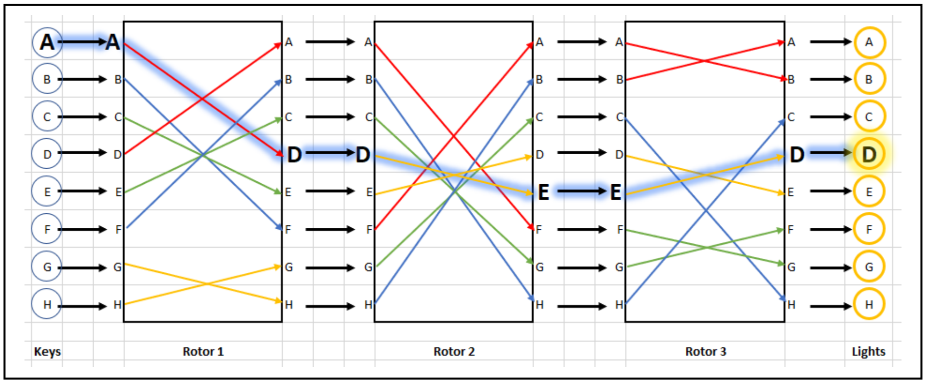
Step 6 – The **E** pin on the right side is in contact with the **E** pin on the left side of the third rotor. This allows the current to move to the third rotor.



Step 7 – The wire connected to the left **E** pin moves the electric current to the **D** pin on the right side of the third rotor.



Step 8 – The **D** pin on the right side is in contact with the **D** light. The electric current is sent to the **D** light, causing it to glow.

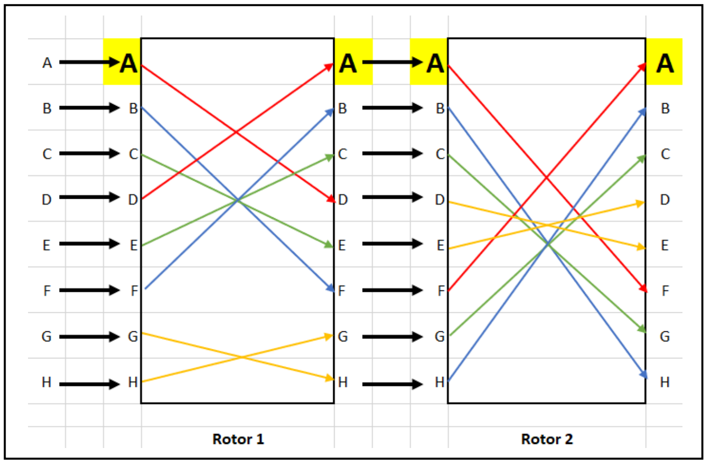


That demonstration illustrated how the Enigma encrypted a single character. But the strength of the rotor machines wasn't the way they encrypted a single character, the strength came from the way they changed the rotors and thus the encryption settings for each character. The way the encryption settings were changed simple to accomplish mechanically, but even though it was simple it provided an encryption scheme that was extremely difficult to crack.

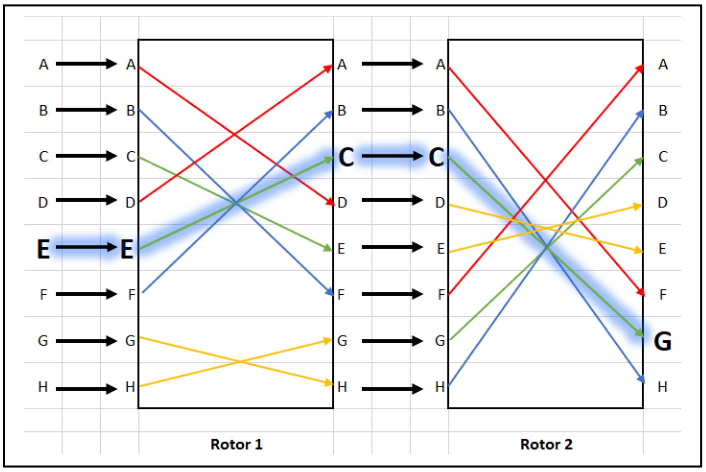
Here's a brief explanation of how the rotors and the encryption settings changed for each character. You just learned how when the operator pressed a key it completed an electric circuit and displayed the encrypted a character by lighting a light. The encryption settings changed when the operator's finger was lifted from the key. At this time the rotors would rotate in a way that's much like the way the hands move on a clock. That is, rotor 1 would behave like the second hand on a clock and rotate every time a character was encrypted. Rotor 2 would behave like the minute hand on a clock, and not move until the second hand, or rotor 1, had made 1 full revolution. Then, when rotor 1 had finished an entire revolution, like a minute hand rotor 2 would advance 1 place. Likewise, rotor 3 would not move until rotor 2 had made 1 complete revolution.

To help you visualize this, the following diagrams illustrate the way the rotors move during the encryption process. Note that in these examples the third rotor has been omitted for clarity. In each step you'll be shown the rotor positions. You'll also see the result of pressing the **E** key in each setting, which path the electric current would take through the rotors and which light would be lit. The **E** key is always used to demonstrate that the encryption settings have changed each time.

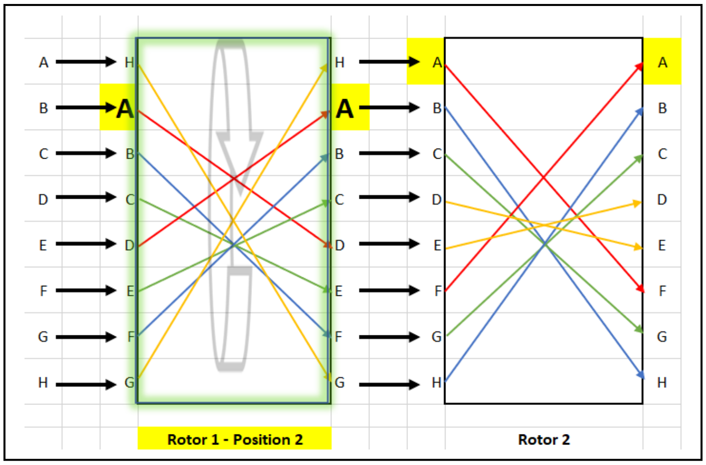
Step 1 – The rotors are in their initial positions.



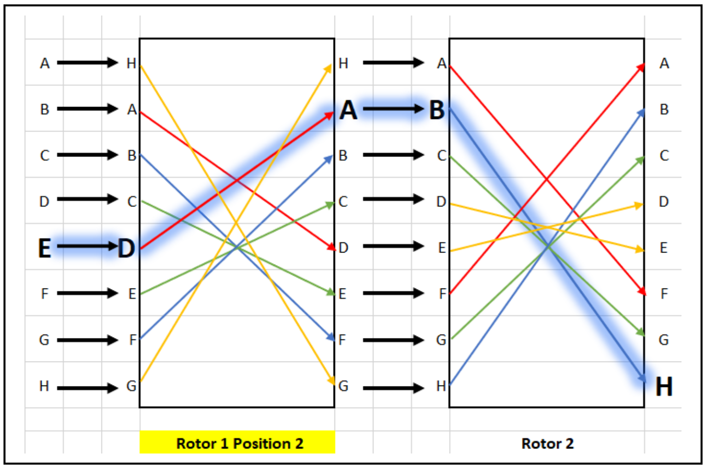
Step 1A – When the **E** key is pressed, the result is the lighting of the **G** light.



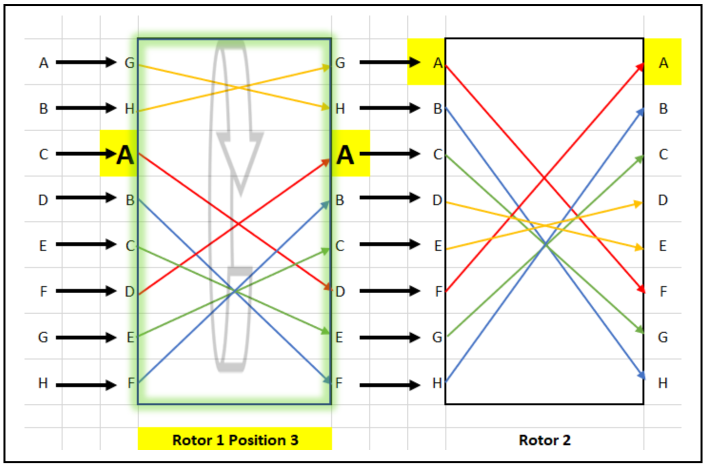
Step 2 – Rotor 1 is rotated 1 position. Notice that all the pins and wires for rotor 1 have shifted. For example, the **A** pin on rotor 1 now touches the **B** key on the left and touches the **B** pin on rotor 2. The **H** wires and pins rotate from the bottom position up to the top position. Also notice that rotor 2 does not move.



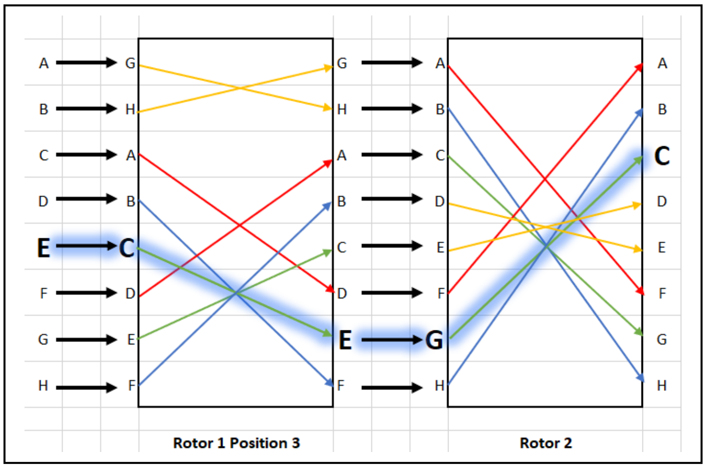
Step 2A – When the **E** key is pressed the electric current will follow a new path through the rotors. With rotor 1 in its new position the result is the **H** light is lit.



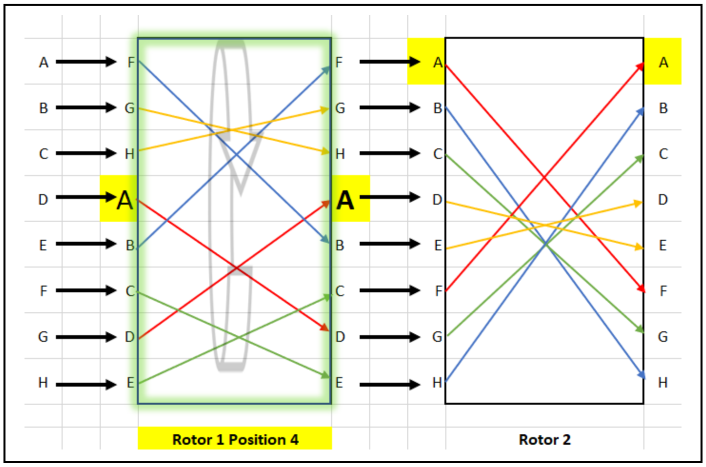
Step 3 – Rotor 1 is once again rotated 1 position. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **C** key on the left and touches the **C** pin on rotor 2. Notice that rotor 2 remains in the same position.



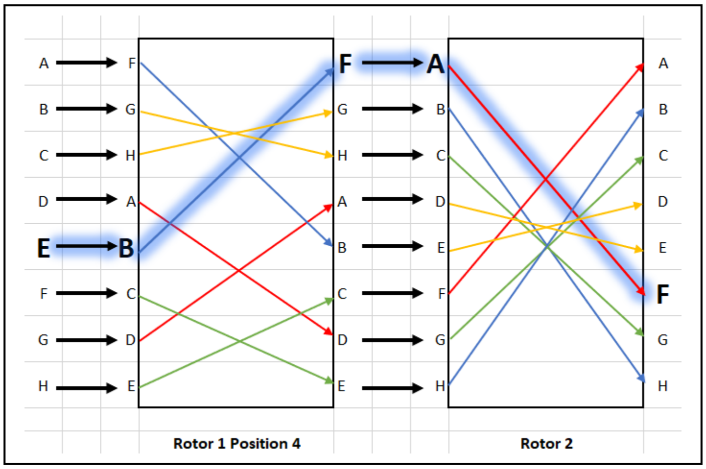
Step 3A – When the **E** key is pressed with rotor 1 in the 3rd position, the result is the **C** light.



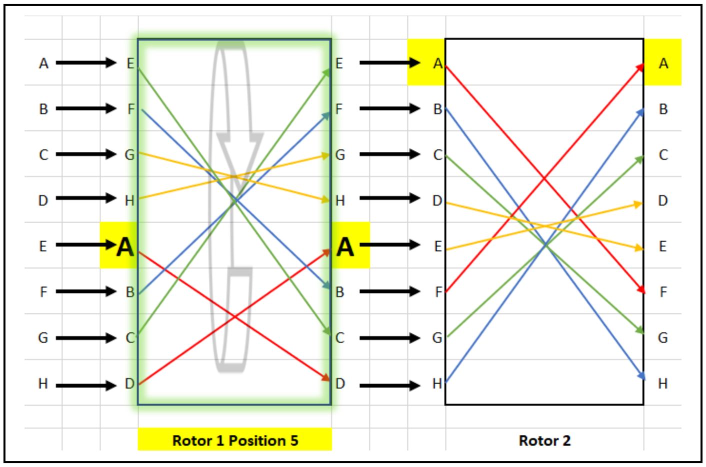
Step 4 – Rotor 1 is rotated once again to the 4th or **D** position. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **D** key on the left and touches the **D** pin on rotor 2. Also note that rotor 2 doesn't move.



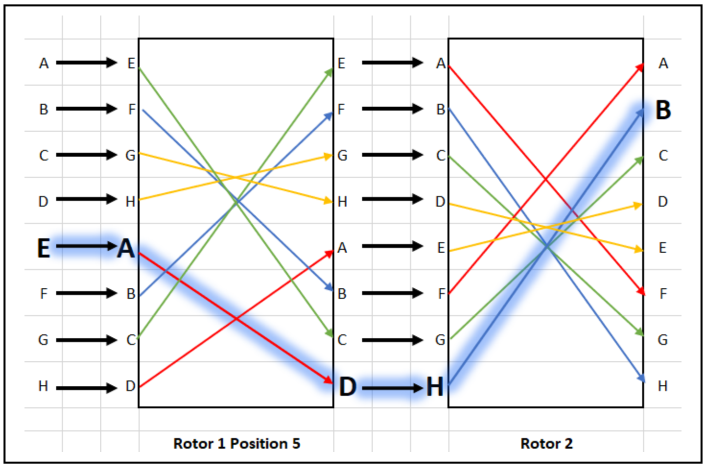
Step 4A – When the **E** key is pressed with rotor 1 in the 4th position, the result is the **F** light.



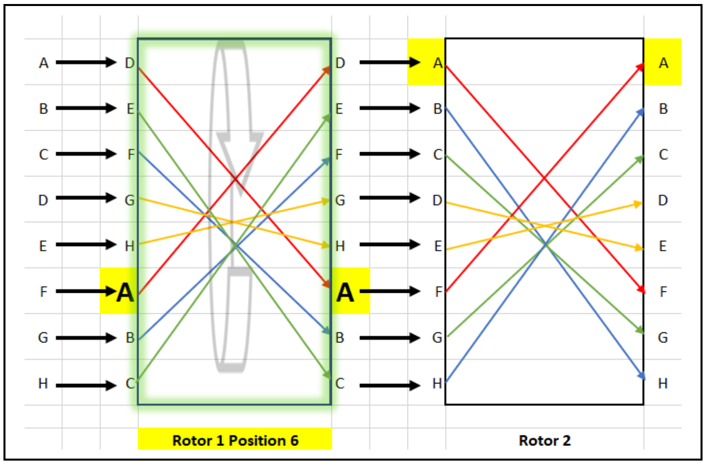
Step 5 – Rotor 1 rotates again to the 5th or **E** position. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **E** key on the left and touches the **E** pin on rotor 2. Rotor 2 still doesn't move.



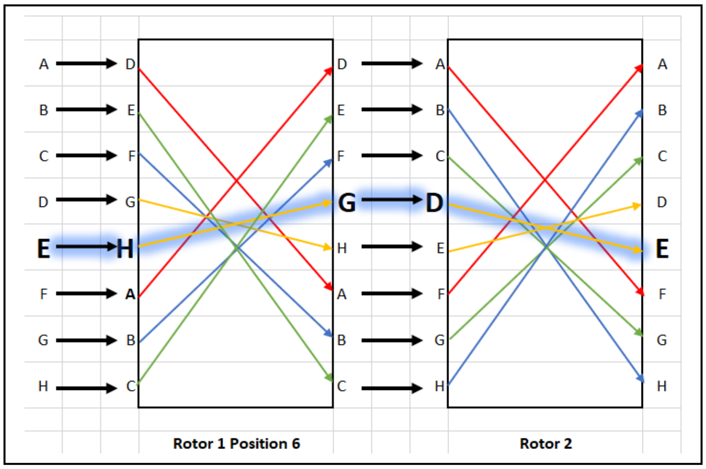
Step 5A – When the **E** key is pressed with rotor 1 in this position, the result is the **B** light.



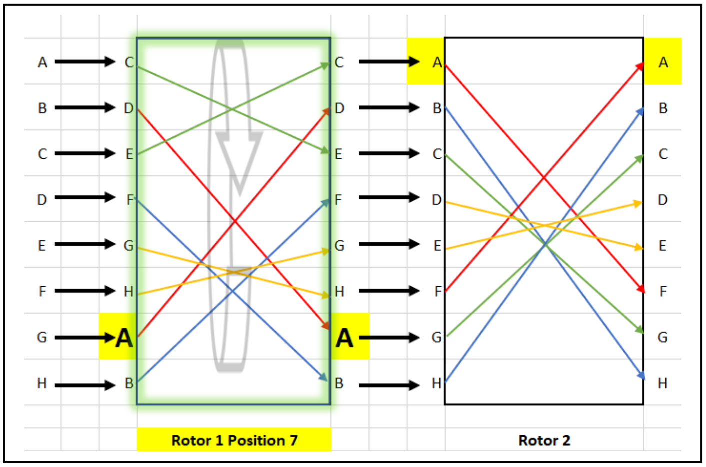
Step 6 – Rotor 1 rotates 1 position again. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **F** key on the left and touches the **F** pin on rotor 2. Also note that rotor 2 is still waiting patiently without moving.



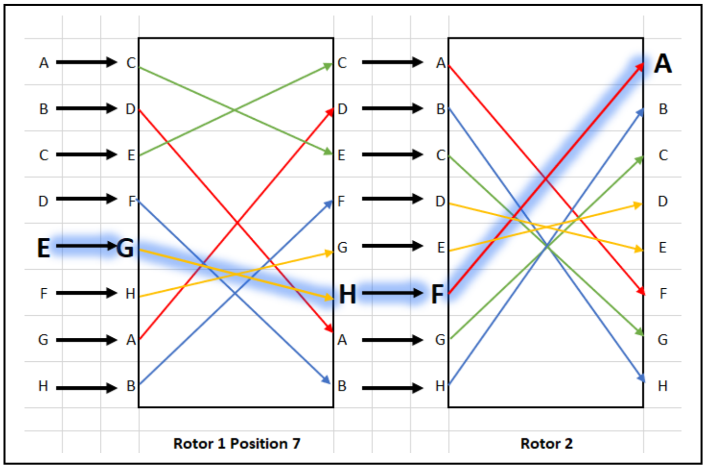
Step 6A – When the **E** key is pressed with rotor 1 in this position, the result is the **E** light.



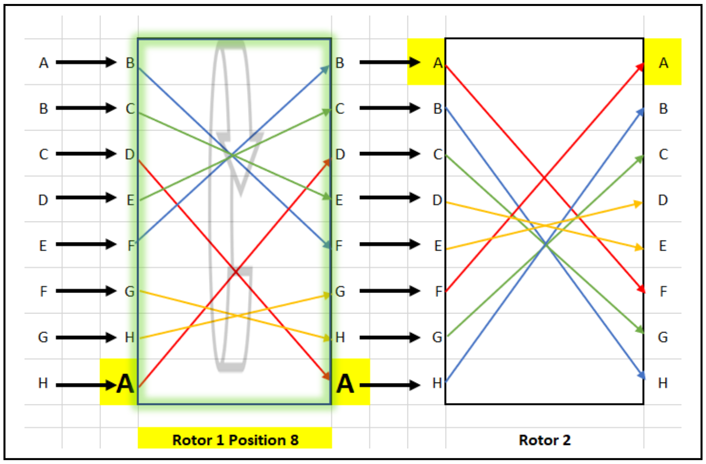
Step 7 – Rotor 1 rotates 1 once again. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **G** key on the left and touches the **G** pin on rotor 2. Also note that rotor 2 still hasn't moved.



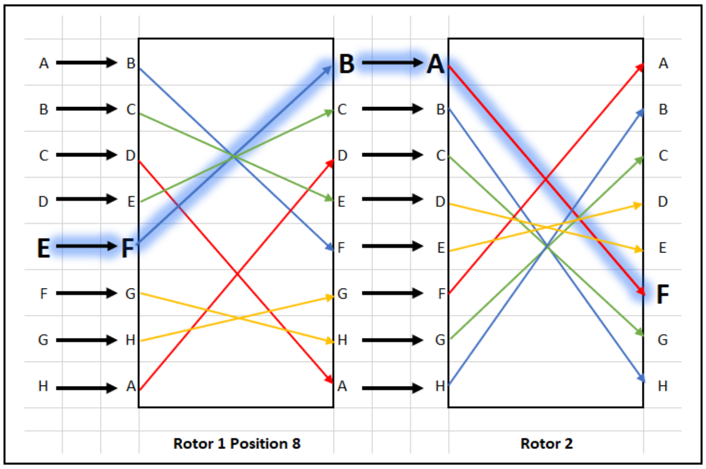
Step 7A – When the **E** key is pressed with rotor 1 in this position, the result is the **A** light.



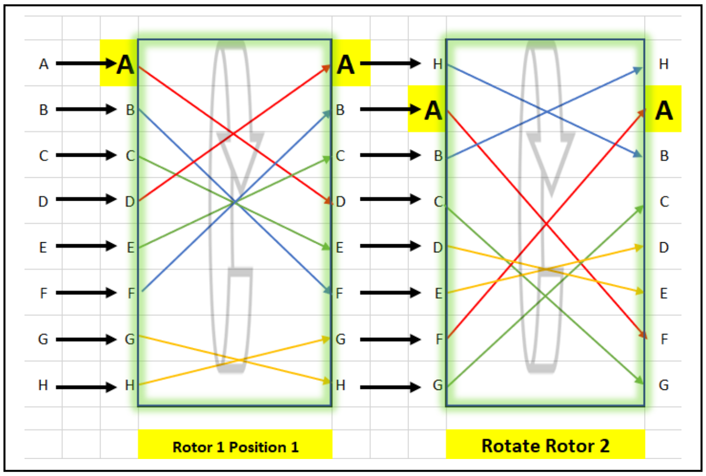
Step 8 – Rotor 1 rotates again. Notice that all the pins and wires for rotor 1 have shifted. Now the **A** pin on rotor 1 now touches the **G** key on the left and touches the **G** pin on rotor 2. Rotor 2 still hasn't moved.



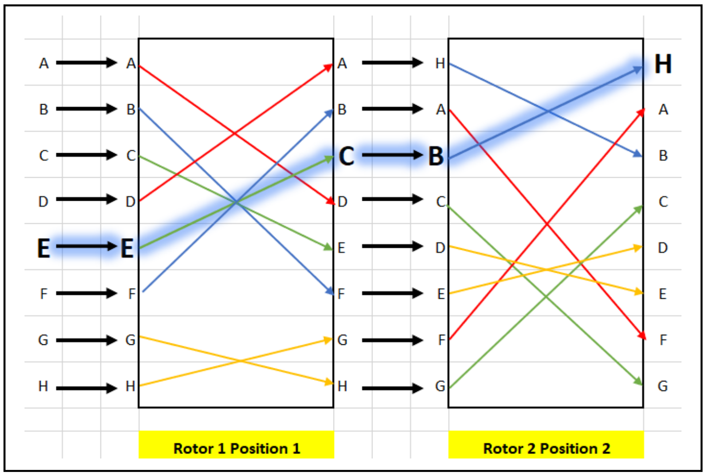
Step 8A – When the **E** key is pressed with rotor 1 in this position, the result is the **F** light. Notice that the **F** light has been lit by the **E** key previously, which is to be expected with this double mapping.



Step 9 – Rotor 1 rotates again. This time, since rotor 1 was in the last position, rotor 2 also rotates 1 position. This means that rotor 1 returns to its starting position where the **A** pin on rotor 1 now touches the **A** key on the left. But since rotor 2 has moved 1 position, the **A** pin on rotor 1 now touches the **H** pin on rotor 2.



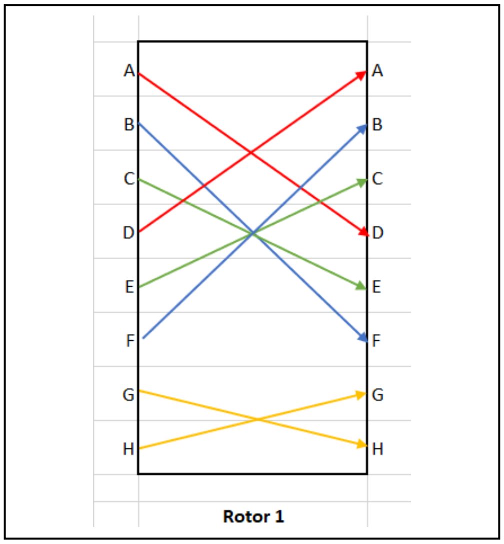
Step 9A – When the **E** key is pressed with the rotor 1 in the 1st position and rotor 2 in the 2nd position the result is the **H** light.



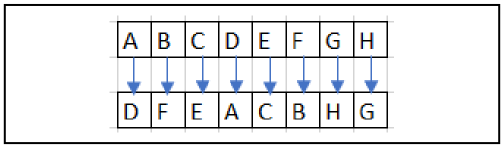
At this point the process repeats, with rotor 1 moving through all 8 positions at which point rotor 2 again advances 1 position. This process will continue, creating unique combinations of character mappings until both rotors have advanced through all 8 possible positions. When the 2 rotors, with 8 positions each, are combined and rotated in this fashion they simulate 64 different substitution ciphers. That is, there are 8 positions for rotor 1 times the 8 positions for rotor 2, and 8x8 is 64.

Rotors as Stacked Substitution Ciphers

To help you put this in perspective and see how it works like a Vigenère cipher on steroids, lets look at it from a slightly different angle. The first thing to do is to show the encryption performed inside rotor 1 as a substitution table. We'll keep using the 8 character rotors for clarity. The figure to the right shows the wiring inside rotor 1 and how it maps the characters.

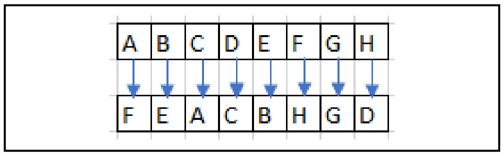


If we rewrite this as a substitution cipher table, while the rotor is in the first position, it will look like the table to the right. Notice that **A** maps to **D**, **B** maps to **F**, etc. just like in rotor 1



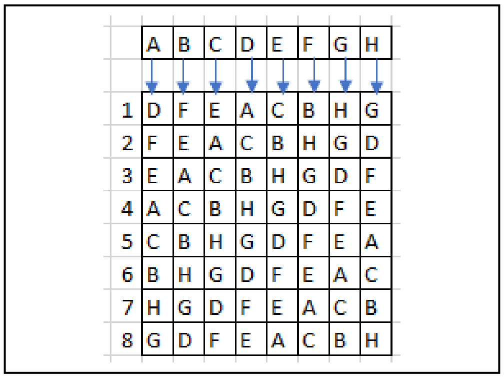
A	B	C	D	E	F	G	H
D	F	E	A	C	B	H	G

When rotor 1 is moved to the 2nd position the substitution table will change, by rotating the cipher text letters in the bottom row of the table 1 place to the left. The result is shown in the figure to the right:



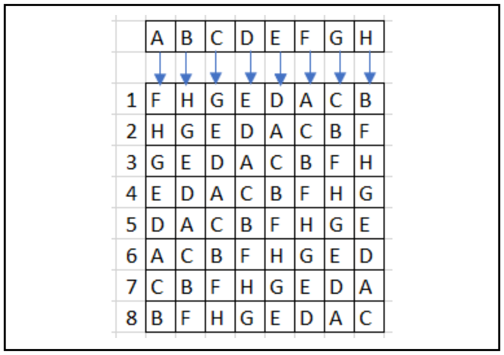
A	B	C	D	E	F	G	H
F	E	A	C	B	H	G	D

This process can be repeated for all the positions of rotor 1, building a table with 8 rows that looks like the table to the right. The resulting table looks like a Vigenère table or tabula recta, except the cipher characters are not in alphabetical order. The cipher text characters rotate from one row to the next just like in the Vigenère table, they're just not in alphabetical order.



	A	B	C	D	E	F	G	H
1	D	F	E	A	C	B	H	G
2	F	E	A	C	B	H	G	D
3	E	A	C	B	H	G	D	F
4	A	C	B	H	G	D	F	E
5	C	B	H	G	D	F	E	A
6	B	H	G	D	F	E	A	C
7	H	G	D	F	E	A	C	B
8	G	D	F	E	A	C	B	H

We can repeat the process of using the wire mappings in rotor 2 to build a second Vigenère like table. The result would be table that looks like the table to the right.

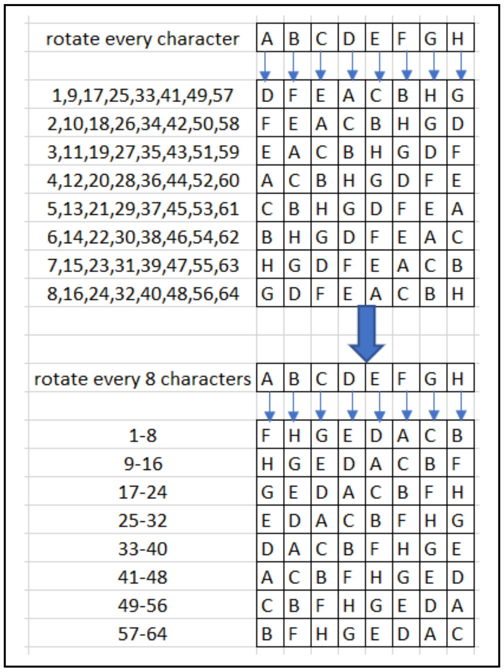


	A	B	C	D	E	F	G	H
1	F	H	G	E	D	A	C	B
2	H	G	E	D	A	C	B	F
3	G	E	D	A	C	B	F	H
4	E	D	A	C	B	F	H	G
5	D	A	C	B	F	H	G	E
6	A	C	B	F	H	G	E	D
7	C	B	F	H	G	E	D	A
8	B	F	H	G	E	D	A	C

The entire encryption process performed by the Enigma machine can be modelled by “stacking” the two tables. By that I mean placing the table for rotor 1 above the table for rotor 2, then taking the output from table 1 and using it as the input for table 2. The thing to note is that the rows for table 1 will change for every input character, while the rows for table 2 will only change after 8 characters have been encrypted.

Once again it’s not important that you absorb every detail of how the Enigma machines encrypt text. But you should start to appreciate how it’s possible to build a non-computerized device to simulate the encryption performed with a Vigenère table.

You should also be able to see that, with this configuration, 64 characters could be encrypted before the substitution pattern repeated. While a Vigenère cipher that used a 64 character could be broken quickly, remember that this has been greatly simplified to make the mechanical encryption process used by the Enigma machines easier to understand.



rotate every character	A	B	C	D	E	F	G	H
1,9,17,25,33,41,49,57	D	F	E	A	C	B	H	G
2,10,18,26,34,42,50,58	F	E	A	C	B	H	G	D
3,11,19,27,35,43,51,59	E	A	C	B	H	G	D	F
4,12,20,28,36,44,52,60	A	C	B	H	G	D	F	E
5,13,21,29,37,45,53,61	C	B	H	G	D	F	E	A
6,14,22,30,38,46,54,62	B	H	G	D	F	E	A	C
7,15,23,31,39,47,55,63	H	G	D	F	E	A	C	B
8,16,24,32,40,48,56,64	G	D	F	E	A	C	B	H

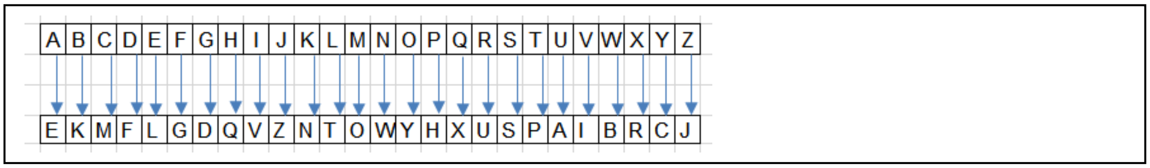
rotate every 8 characters	A	B	C	D	E	F	G	H
1-8	F	H	G	E	D	A	C	B
9-16	H	G	E	D	A	C	B	F
17-24	G	E	D	A	C	B	F	H
25-32	E	D	A	C	B	F	H	G
33-40	D	A	C	B	F	H	G	E
41-48	A	C	B	F	H	G	E	D
49-56	C	B	F	H	G	E	D	A
57-64	B	F	H	G	E	D	A	C

The Real Enigma

The last thing you need to know about the Enigma machines is that they had several features that I simplified or left out in previous explanation. These features all contributed to making encryption produced by the Enigma much harder to crack. These features include the following:

- The first and most obvious is that the rotors had 26 characters instead of 8. This meant that with 3 rotors there were 26x26x26 or 17,576 different mappings, or with 4 rotors 26^4 or 456,976 mappings.

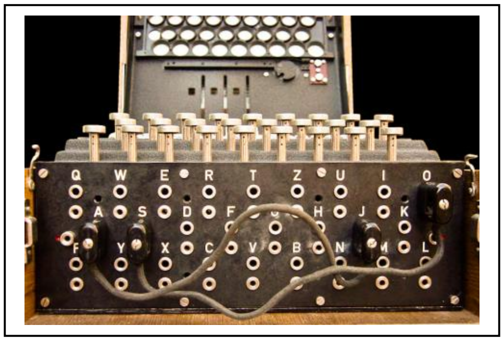
- In the examples rotor wiring diagrams the character mappings are mirrored. That is, if **A** maps to **C** then **C** maps to **A**. The real rotors mappings didn’t do this. It may have happened by chance, but mirroring the characters was not a rule. The following figure shows the real mappings from one of the actual Enigma rotors. Note that **A** maps to **E**, but **E** maps to **L**.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
E	K	M	F	L	G	D	Q	V	Z	N	T	O	W	Y	H	X	U	S	P	A	I	B	R	C	J

- The rotors could be removed and replaced in a different order. That is, the order could be 1-2-3 or 2-1-3. With 3 rotors there are 6 possible ways to configure the rotors. This was typically done once a day as part of the daily configuration.

- The Enigma machines had a plug board, or patch panel, on the front. This patch panel allowed characters to be manually swapped before they entered the rotors. Once again, this acted like a rotor, except once set, the mappings wouldn't change without manual intervention. Originally only 3 plugs were used, but this was later increased to 6 plugs. The plugs were typically set once each day at the same time the rotors were set. But, unlike the rotors, the character swaps performed by the plug board could be changed at will.



  The ability to change the configuration of the plug board provided a tremendous amount of variation and thus strength to the overall cipher. If you step back and think about it, the plug board acted like a substitution cipher. If the substitution cipher were swapping all 26 characters the number of variations would be 26! But since the Enigma used 6 plugs this number was decreased to 100,391,791,500. By itself this substitution cipher would be very easy to break with frequency analysis. But combining it with the rest of the Enigma encryption easily flattened the frequency graphs and made it very difficult to break.

[i] Photo from: https://www.manchester.ac.uk/discover/news/enigma-machine-visits-the-alan-turning-building/

[ii] Photo from: https://www.sothebys.com/en/articles/breaking-the-code-the-secrets-of-enigma-cipher-machines

[iii] Photo from: https://www.geocaching.com/track/details.aspx?id=3640554

[iv] Photo By Bob Lord - German Enigma Machine, uploaded in english wikipedia on 16. Feb. 2005 by en:User:Matt Crypto, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=258976