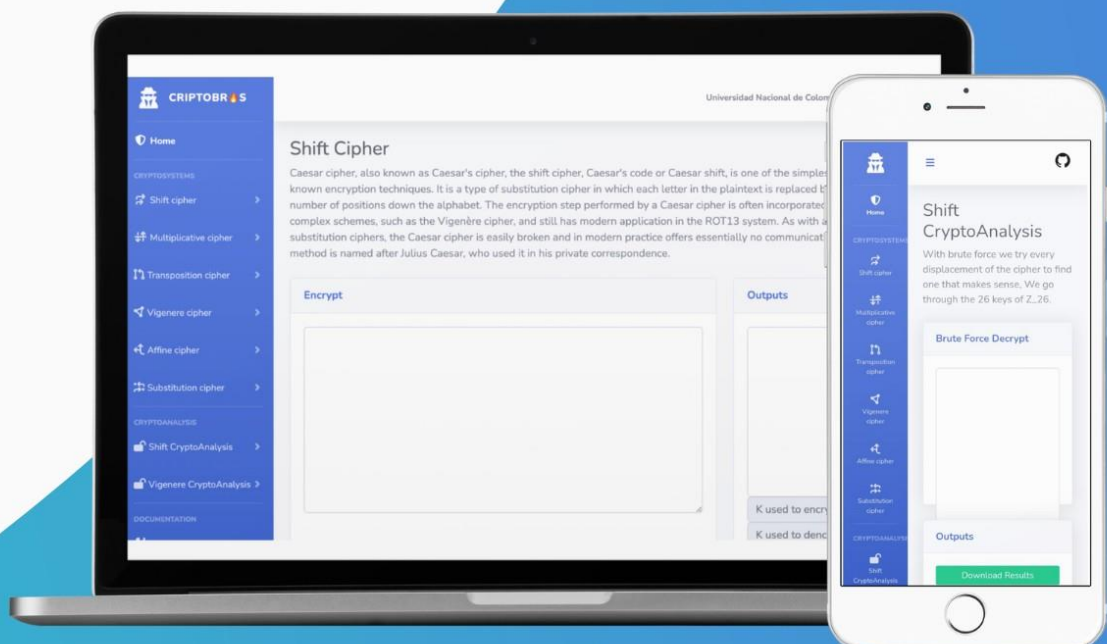




CRIPTOBROS

CRIPTOBROS USER MANUAL



Content

1.	Introduction	1
2.	CryptoBros Online App Overview	2
3.	Cryptosystems	4
	Shift Cipher System	4
	Multiplicative Cipher System	5
	Transposition Cipher System	7
	Vigenère Cipher System	8
	Affine Cipher System.....	10
	Substitution Cipher System.....	11
	Hill Image System.....	13
4.	Crypto Analysis	15
	Shift Cryptanalysis	15
	Vigenère Cryptanalysis.....	16
	Affine Cryptanalysis.....	16
5.	Block cipher Cryptosystems	18
	SDES cipher	18
	DES cipher	19
	DESimage cipher.....	21
	AESimage cipher	22
	TDESimage cipher	24
	Gamma-Pentagonal Cipher	25
6.	Public-key Cryptosystems	28
	RSA Cryptosystem	29
	The Rabin Cryptosystem.....	29
	ElGamal Cryptosystem	30
7.	Page Design	35
	Templates	35
	Colors51	36

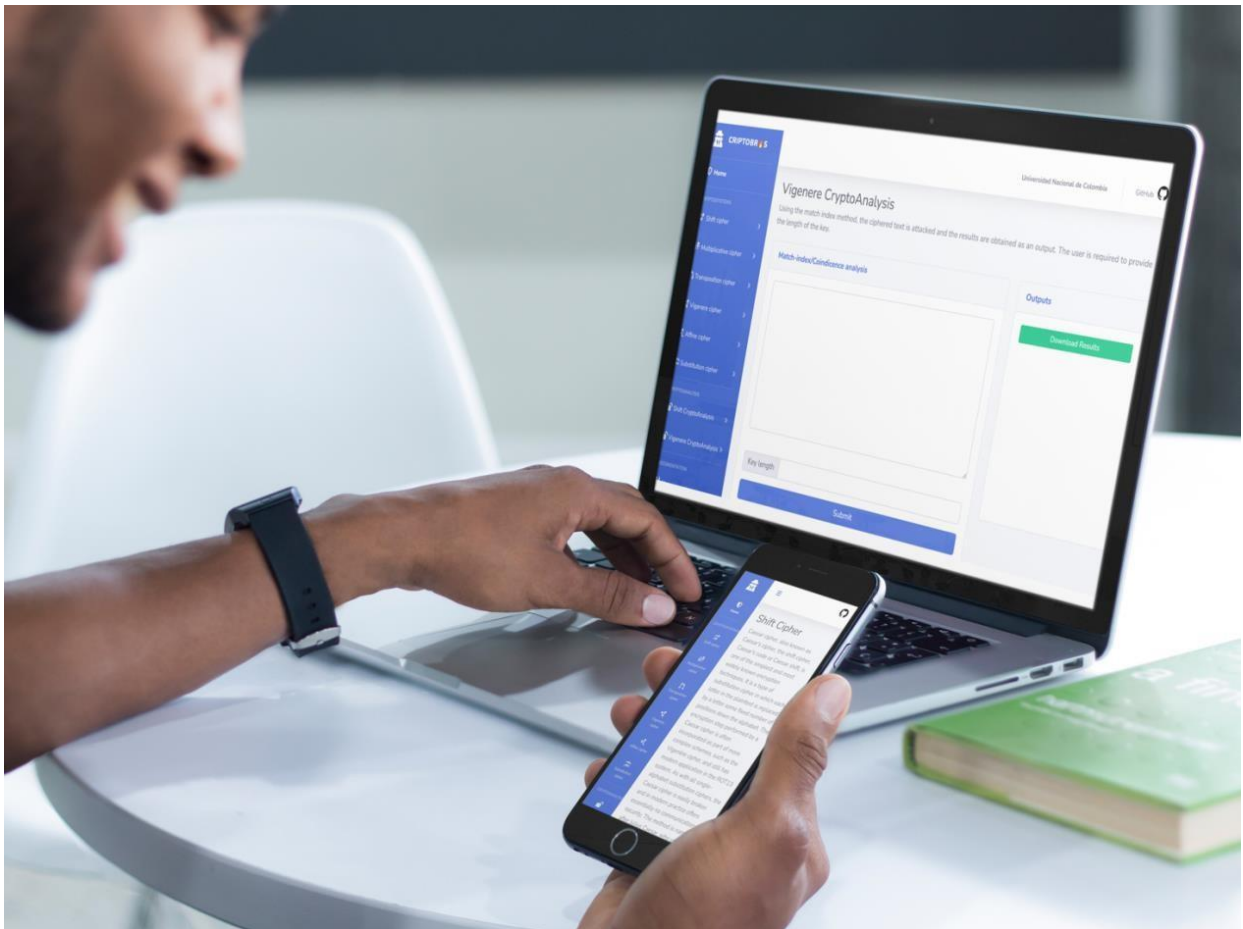
1. Introduction:

Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in actual practice by any adversary. While it is theoretically possible to break into a well-designed system, it is infeasible in actual practice to do so.

CriptoBros allows you to easily encrypt and decrypt plain text using an online application. The product includes everything necessary to have a complete presence on the Internet:

- the online encryption and decryption creation tool (the operation of which is explained in detail in this manual).

The application is intuitive and easy to use: you do not need to know cryptography, just have the texts that you want to encrypt or analyze. They adapt to any device (PCs, Tablets, and Smartphones) without the need for any extra steps.



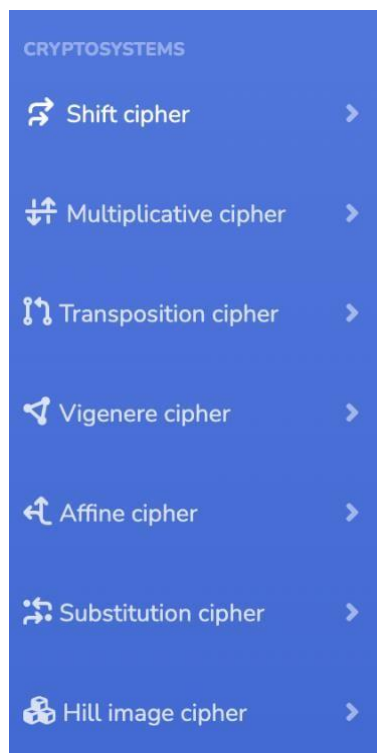
2. CriptoBros Online App Overview:

The panel of CriptoBros is divided into 4 areas:

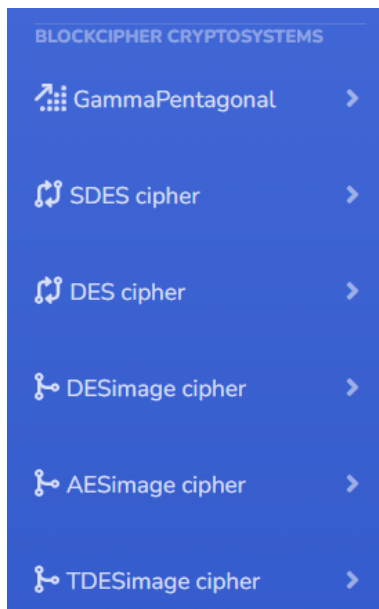
1. **Home:** We can see the name of the page **CriptoBros**, and some other features like the name of the University, Universidad Nacional de Colombia, that will carry you to the principal page of the university with just one click away, right next to it, will be found a logo of GitHub where you will be founding all the frontend code and backend code of the page.



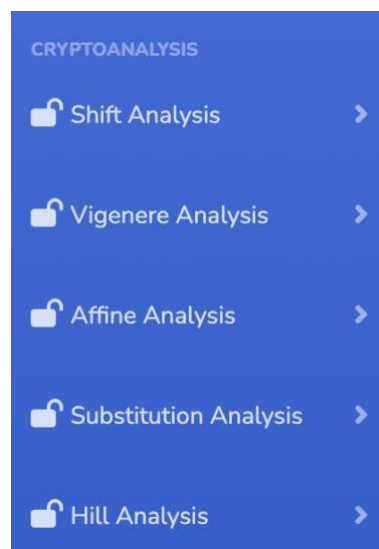
2. **Cryptosystems:** Suite of cryptographic algorithms needed to implement a particular security service, such as confidentiality. for example, we have: Shift cipher, Multiplicative cipher, Transposition cipher, Vigenère cipher, Affine cipher, and Substitution cipher.



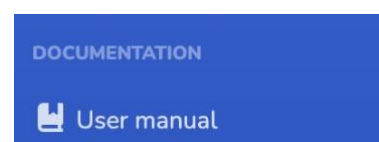
3. **Block cipher Cryptosystems:** A block cipher by itself allows encryption only of a single data block of the cipher's block length. For a variable-length message, the data must first be partitioned into separate cipher blocks. In the simplest case, known as electronic codebook (ECB) mode, a message is first to split into separate blocks of the cipher's block size (possibly extending the last block with padding bits), and then each block is encrypted and decrypted independently.



4. **Cryptoanalysis:** It is used to breach cryptographic security systems and gain access to the contents of encrypted messages, even if the cryptographic key is unknown. We can find some of them as Substitution cryptoanalysis, Shift cryptoanalysis, Vigenère cryptoanalysis, Affine cryptoanalysis, and Hill cryptoanalysis.



5. **Documentation:** This provides easily accessible information on this product and gives answers to important questions about product usage in general. aspects of functionality. The architecture of a technicalproduct.



3. Crypto Systems:

Let's take a view of each of the cryptosystems:

1. **Shift Cipher System:** The first to see it's a is a brief explanation of the type of system, like: "It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet". Then you can find 3 boxes:

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text.

It is also needed a number key (K) between 1-26, in the box under the one you put the plain text so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random one with just pressing submit.

- **Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.

Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

K used to encrypt	12
K used to dencrypt	

- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using the key (K) between 1-26 mentioned before.

Decrypt

K

Submit

2. **Multiplicative Cipher System:** The first to see it is a brief explanation of the type of system, like: "Multiplicative Ciphers work by using the modulo operator to encrypt and decrypt messages." Then you can find 3 boxes:

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text.

Encrypt

My secret message

It is also needed a prime number key (K) module 26 in the box under the one you put

the plain text, so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random prime number key (K) module 26 by just pressing submit.

K

Submit

- Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.

Outputs

IQMUKHURIUMMAEU

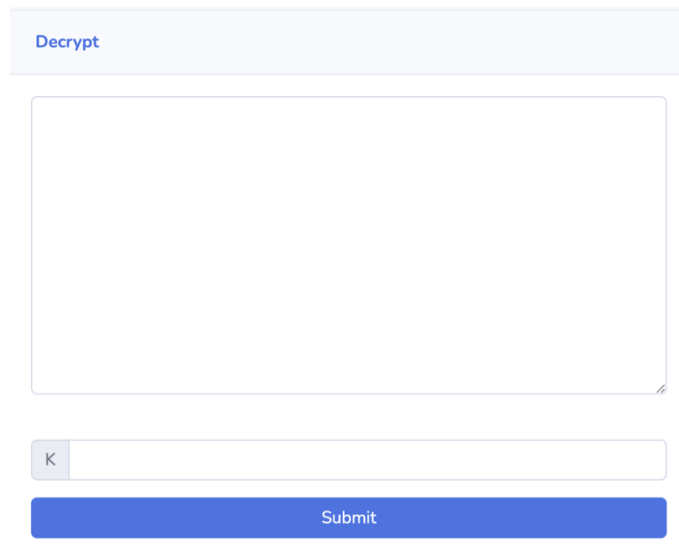
Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

K used to encrypt

5

K used to decrypt

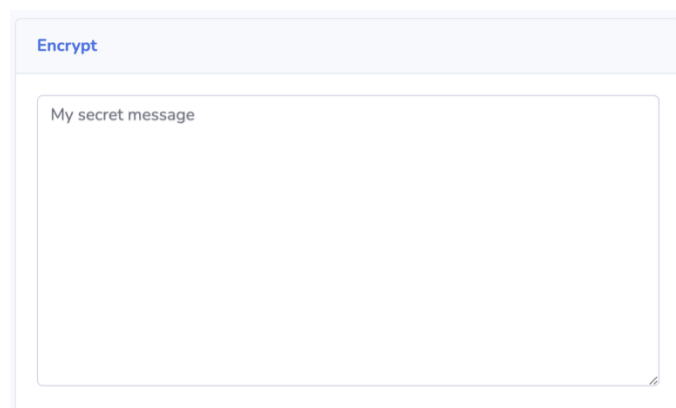
- Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using the prime number key (K) module 26 mentioned before.



The Decrypt interface consists of a light blue header bar with the word "Decrypt" in blue. Below the header is a large, empty white rectangular box for input. Underneath this box is a key input section with a small grey box containing the letter "K" and a white text input field. At the bottom of the interface is a solid blue button with the word "Submit" in white text.

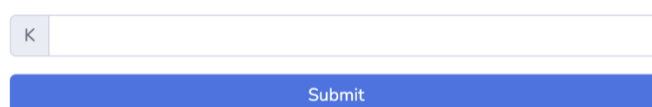
3. **Transposition Cipher System:** The first to see it's a is a brief explanation of the type of system, like: "Transposition cipher is the name given to any encryption that involves rearranging the plain text letters in a new order." Then you can find 3 boxes:

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text.



The Encrypt interface has a light blue header bar with the word "Encrypt" in blue. Below the header is a large white rectangular box containing the placeholder text "My secret message". Below this box is a key input section with a small grey box containing the letter "K" and a white text input field. At the bottom of the interface is a solid blue button with the word "Submit" in white text.

It is also needed a number key (K) between 2 and the length of the plain text in the box under the one you put the plain text so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random key (K) with just pressing submit.



This block shows a close-up of the key input section and the submit button. It features a small grey box with the letter "K" next to a white text input field. Below these is a solid blue button with the word "Submit" in white text.

- **Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.



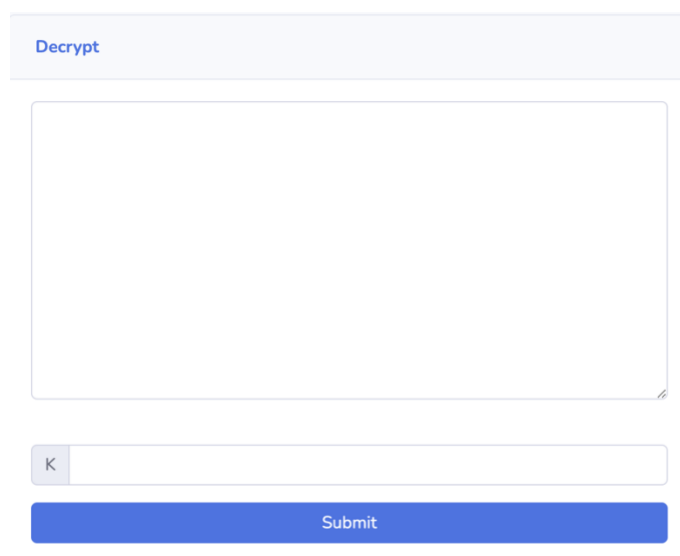
The screenshot shows a light blue header with the word "Outputs". Below it is a large white rectangular box containing the text "MCMAYREGSESEETS".

Under that box it will be shown the key (K) used to encrypt and decrypt other it's encrypted or decrypted.



The screenshot shows two input fields. The first field is labeled "K used to encrypt" and contains the number "4". The second field is labeled "K used to decrypt" and is currently empty.

- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using a number key (K) between 2 and the length of the plain text mentioned before.



The screenshot shows a light blue header with the word "Decrypt". Below it is a large white rectangular box for entering the encrypted text. At the bottom, there is a small input field labeled "K" and a blue "Submit" button.

4. **Vigenère Cipher System:** The first to see it is a brief explanation of the type of system, like: "Vigenère Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets." Then you can find 3 boxes:

- **Encrypt:** Where you put your top-secret message or just something you want to

say in plain text.

Encrypt

My secret message

It is also needed a string key (K), which can be a string with a length between 1 and the length of the plain text, in the box under the one you put the plain text, so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random key (K) with the specifications mentioned before by just pressing submit.

K

Submit

- Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.

Outputs

GKXJVVYFRJLXUSJ

Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

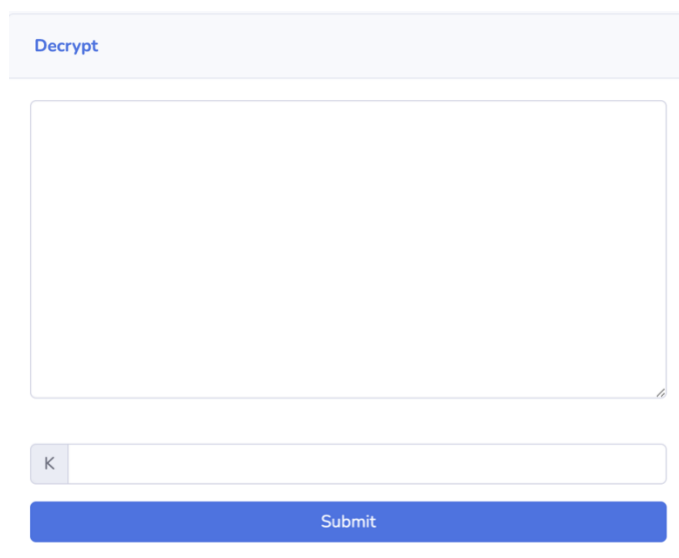
Keyword used to encrypt

UM
FFT
F

Keyword used to decrypt

- Decrypt:** If you have an encrypted text, use the decryption box to return to the

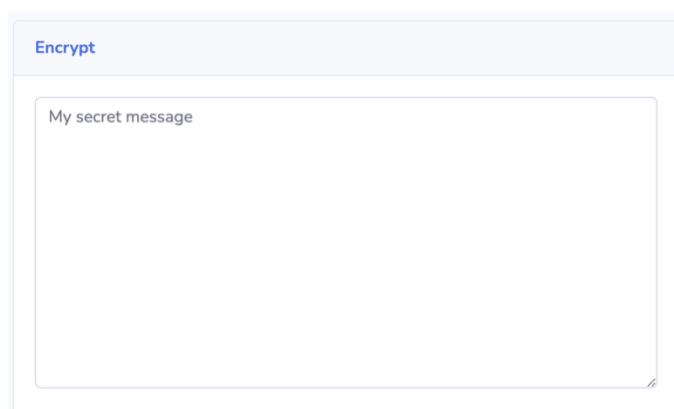
plain text the string key (K) mentioned before.



The image shows a web interface for decryption. At the top, there is a light blue header with the word "Decrypt" in a darker blue font. Below the header is a large, empty rectangular text box with a thin border and a small cursor icon at the bottom right. Underneath the text box is a small input field with a grey background and the letter "K" on the left, followed by a white text input area. At the bottom of the interface is a solid blue button with the word "Submit" in white text.

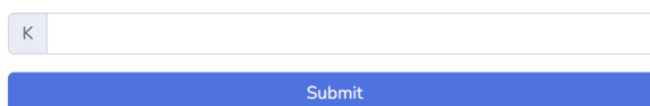
5. **Affine Cipher System:** The first to see it is a brief explanation of the type of system, like: "The affine cipher is a type of monoalphabetic substitution cipher, where each letter in an alphabet is mapped to its numeric equivalent, encrypted using a simple mathematical function, and converted back to a letter." Then you can find 3 boxes:

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text.



The image shows a web interface for encryption. At the top, there is a light blue header with the word "Encrypt" in a darker blue font. Below the header is a large rectangular text box with a thin border. The text "My secret message" is entered at the top of this box, and a small cursor icon is at the bottom right. Below the text box is a small input field with a grey background and the letter "K" on the left, followed by a white text input area. At the bottom of the interface is a solid blue button with the word "Submit" in white text.

It is also needed a key (K), which can be a tuple that is relative numbers mod 26, in the box under the one you put the plain text, so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random key (K) with the specifications mentioned before by just pressing submit.



The image shows a key input field. It consists of a small grey box with the letter "K" on the left, followed by a white text input area. Below this is a solid blue button with the word "Submit" in white text.

- **Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.

Outputs

ZTJDRQDCZDJJFPD

Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

K used to encrypt	19,5
K used to decrypt	

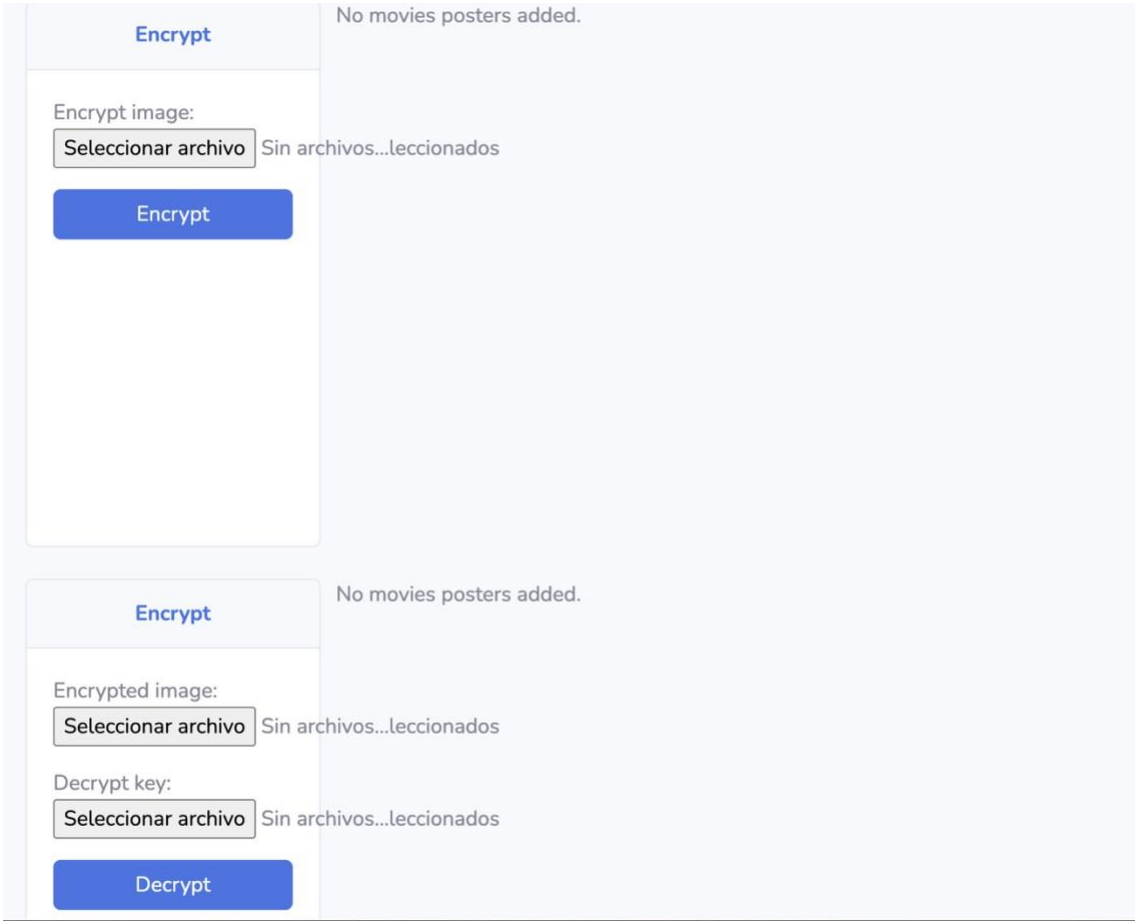
- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using the tuple that are relative numbers mod 26 key (K) mentioned before.

Decrypt

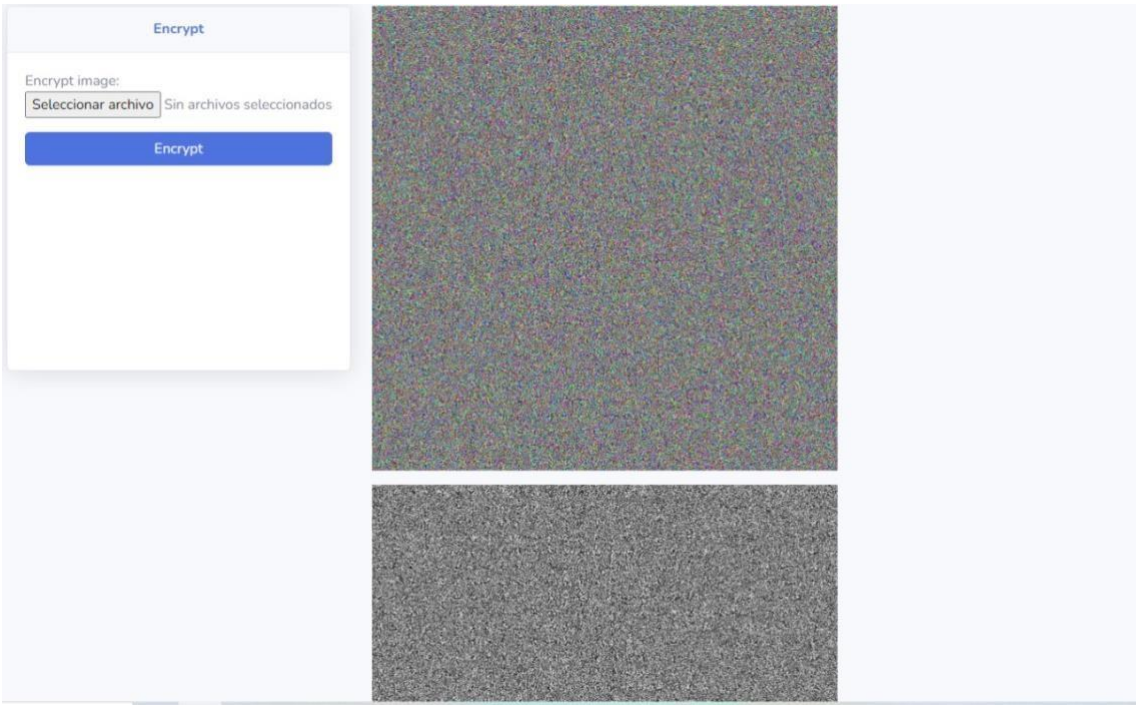
K

Submit

6. **Hill Image System:** The Hill cipher algorithm is a symmetric key algorithm with several advantages in data encryption. But, the inverse of the key matrix used for encrypting the plaintext does not always exist. Then if the key matrix is not invertible, encrypted text cannot be decrypted. In the Involutory matrix generation method, the key matrix used for the encryption is itself invertible.

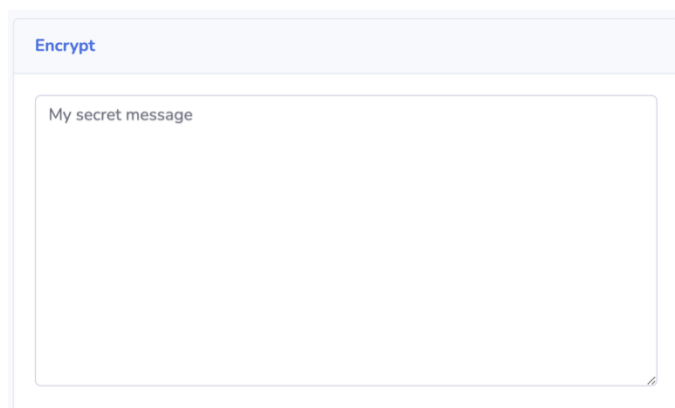


Implemented Hill Cipher technique one is a cover image which acts as a key image which is shared by both sender and receiver and other is Informative image. As the first step, we add a cover image and informative image to obtain the resultant image. Uploading an image

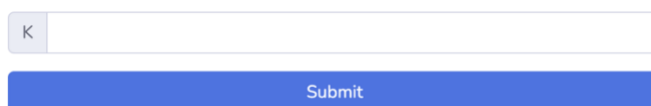


7. **Substitution Cipher System:** The first to see it is a brief explanation of the type of system, like: "Substitution cipher is a method of encrypting in which units of plaintext are replaced with the ciphertext, in a defined manner, with the help of a key; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth." Then you can find 3 boxes:

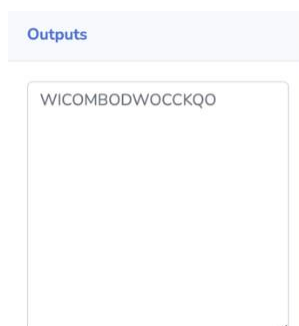
- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text.



It is also needed a number key (K, so that it can encrypt the message, or if you don't want to give any key, it will be provided by a random key (K) just pressing submit.



- **Outputs:** You will find the encrypted or the plain text. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., plain text.



Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

K used to encrypt	95459972
K used to decrypt	

- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using a key (K) mentioned before.

Decrypt

K

Submit

4. Crypto Analysis:

Cryptanalysis is the process of studying cryptographic systems to look for weaknesses or leaks of information. Let's see how we can break in.

1. **Shift Cryptanalysis:** With brute force, we try every displacement of the cipher to find one that makes sense, we go through the 26 keys of Z_26.

Brute Force Decrypt

Submit

In the box shown above, we write the encrypted message we want to decrypt, then by pressing submit we get a result we will be downloading, giving us a .txt file with decrypted code.

Outputs

Download Results

```

SHIFT CRIPTO ANALYSIS
CIPHER TEXT: LXRDBQDSLDRZFD
-----
With key 0, the clear text is LXRDBQDSLDRZFD
With key 1, the clear text is KMOCAPCRKCOQYEC
With key 2, the clear text is JVPBQBOQJBPPXDB
With key 3, the clear text is IUDAYNAPIADQWCA
With key 4, the clear text is HTNZXHZOHZNVVBZ
With key 5, the clear text is GSHYWLNYGYMHUAY
With key 6, the clear text is FRLXVKQMFLLTZX
With key 7, the clear text is EGRWJWLEWKSXYW
With key 8, the clear text is DPJYTIWKDVJJRSV
With key 9, the clear text is COIUSHUJCUIIQWU
With key 10, the clear text is BNHTRGTBTHHPVT
With key 11, the clear text is AMGSQFSHASGGOUS
With key 12, the clear text is ZLFRPERGZRFNTR
With key 13, the clear text is YKEODDQPYOEEHSD
With key 14, the clear text is XJDPNCPXPQDLRP
With key 15, the clear text is WICOMBODMOCKQO
With key 16, the clear text is VHBNLANCNVNBJPV
With key 17, the clear text is UGAMKZMBUMAAIOM
With key 18, the clear text is TFZLJYLALZZHNL
With key 19, the clear text is SEYKXKZSKYQMK
With key 20, the clear text is RDXJHJYRJXXFLJ
With key 21, the clear text is QCWIGVIXQIWWEKI
With key 22, the clear text is PBVHFUHWPHVVDJH
With key 23, the clear text is OAUGETGVGOUUCIG
With key 24, the clear text is NZTFDSFUNFTTBWF
With key 25, the clear text is MYSECRETMESAGE
    
```

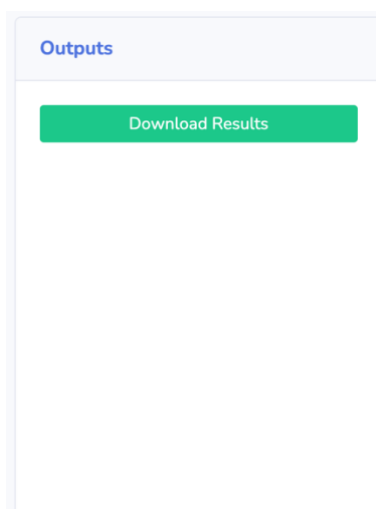
As you can see, the decrypted code was in key 25, as it says, my secret message.

2. **Vigenère Cryptanalysis:** Using the match index method, the ciphered text is attacked and the results are obtained as an output. The user must provide the key's length and press Submit.



The screenshot shows a web interface titled "Match-index/Coincidence analysis". It features a large, empty text input box for entering a message. Below this box is a smaller input field labeled "Key length". At the bottom of the interface is a blue button labeled "Submit".

In the box shown above, we write the encrypted message we want to decrypt, by pressing submit we get a result we will be downloading, giving us a .txt file with the possible key and the decrypted code.



The screenshot shows a web interface titled "Outputs". It contains a single green button labeled "Download Results".

3. **Affine Cryptoanalysis:** With brute force, we try every possible combination of keys to find one that makes sense. We go through the 312 possible keys. And that is because there are finite combinations of a and b .

Match-index/Coincidence analysis

Key length

Submit

In the box shown above, we write de encrypted message we want to decrypt, by pressingsubmit we get the result, and will be downloaded, giving a .txt file

Outputs

Download Results

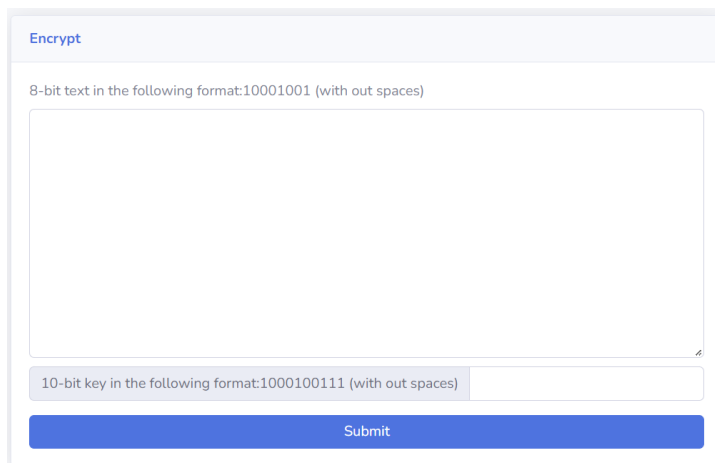
Assuming the keys are 17 and 0, the text is	DPJVTIVKDVJJRXV
Assuming the keys are 17 and 1, the text is	GSMYWLNGYMMUAY
Assuming the keys are 17 and 2, the text is	JVPBZOBQJBPPXDB
Assuming the keys are 17 and 3, the text is	MYSECRETMESSAGE
Assuming the keys are 17 and 4, the text is	PBVHFUHWPHVVDJH
Assuming the keys are 17 and 5, the text is	SEYKIXKZSKYYGMK
Assuming the keys are 17 and 6, the text is	VHBNLANCVNBBJPN
Assuming the keys are 17 and 7, the text is	YKEQODQFYQEEMSQ

5. Block cipher Cryptosystems:

A block cipher uses a symmetric key and algorithm to encrypt and decrypt a block of data. A block cipher requires an initialization vector (IV) that is added to the input plaintext to increase the key space of the cipher and make it more difficult to use brute force to break the key.

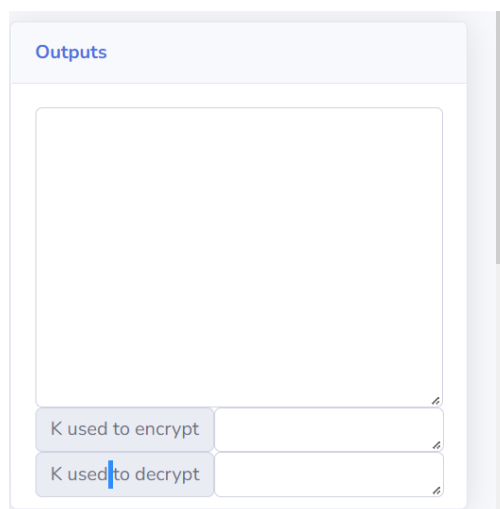
- 1. SDES Cipher:** It is a simple version of the DES Algorithm. It is similar to the DES algorithm but is a smaller algorithm and has fewer parameters than DES. It was made for educational purposes so that understanding DES would become simpler. It is a block cipher that takes a block of plain text and converts it into ciphertext. It takes a block of 8 bit:

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text with a 10-bit key in the following format:1000100111 (without spaces)



The 'Encrypt' interface features a title bar with the word 'Encrypt' in blue. Below the title bar, there is a text prompt: '8-bit text in the following format:10001001 (with out spaces)'. This is followed by a large, empty text input area. Below the input area, there is a text prompt: '10-bit key in the following format:1000100111 (with out spaces)' next to a smaller text input field. At the bottom of the interface is a prominent blue button labeled 'Submit'.

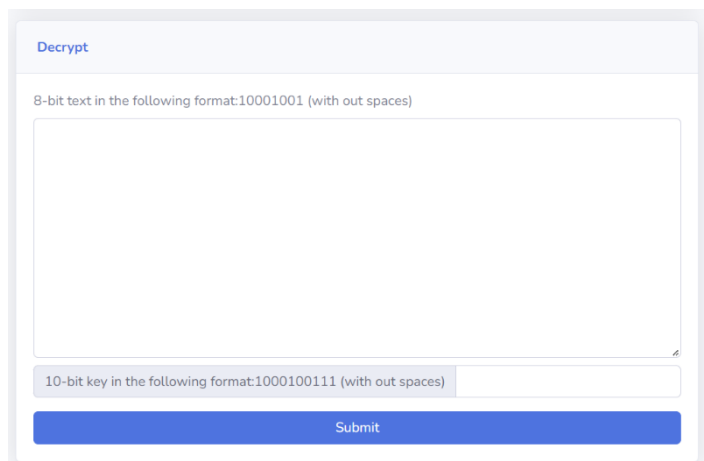
- **Outputs:** You will find the encrypted or the plain text with 8-bit text in the following format:10001001. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., 8-bit text in the following format:10001001.



The 'Outputs' interface has a title bar with the word 'Outputs' in blue. Below the title bar is a large, empty text area for displaying results. At the bottom of the interface, there are two rows of labels and input fields. The first row is labeled 'K used to encrypt' and the second row is labeled 'K used to decrypt', each followed by an empty text input field.

Under that box it will be shown de key (K) used to encrypt and decrypt other it's encrypted or decrypted.

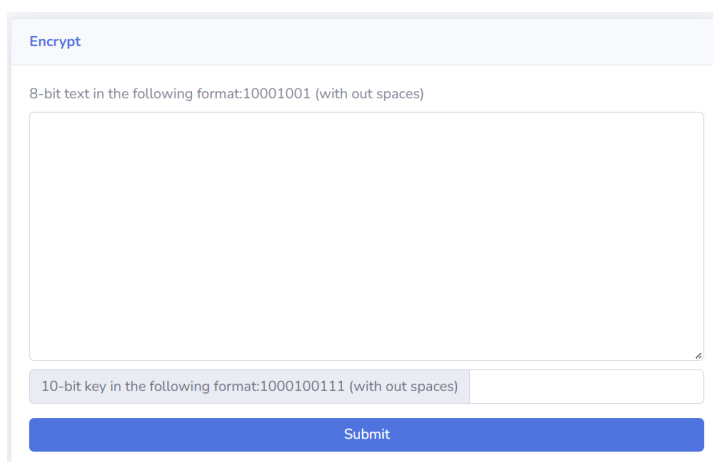
- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using a key (K) mentioned before.



The screenshot shows the 'Decrypt' interface. At the top, the word 'Decrypt' is displayed in a small blue font. Below it, a label reads '8-bit text in the following format:10001001 (with out spaces)'. There is a large, empty rectangular text input box. Below this box, another label reads '10-bit key in the following format:1000100111 (with out spaces)', followed by a smaller, empty rectangular text input box. At the bottom of the interface is a blue button with the word 'Submit' in white text.

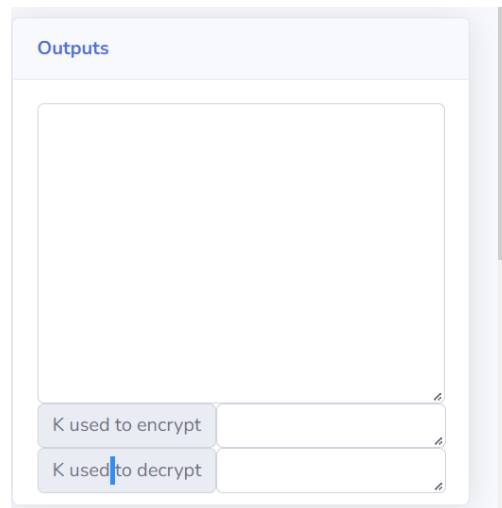
2. **DES Cipher:** It is a symmetric-key algorithm for the encryption of digital data. Although its short key length of 56 bits makes it too insecure for modern applications, it has been highly influential in the advancement of cryptography.

- **Encrypt:** Where you put your top-secret message or just something you want to say in plain text with a 10-bit key in the following format:1000100111 (without spaces)



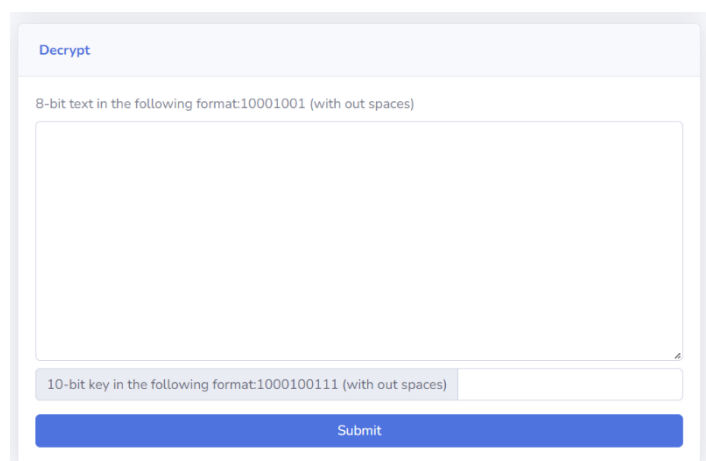
The screenshot shows the 'Encrypt' interface. At the top, the word 'Encrypt' is displayed in a small blue font. Below it, a label reads '8-bit text in the following format:10001001 (with out spaces)'. There is a large, empty rectangular text input box. Below this box, another label reads '10-bit key in the following format:1000100111 (with out spaces)', followed by a smaller, empty rectangular text input box. At the bottom of the interface is a blue button with the word 'Submit' in white text.

- **Outputs:** You will find the encrypted or the plain text with 8-bit text in the following format:10001001. If you use the Encrypt box you will find the encrypted text, on the other hand, if you use the Decrypt box you will see the decrypted text, i.e., 8-bit text in the following format:10001001.



Under that box it will be shown the key (K) used to encrypt and decrypt other it's encrypted or decrypted.

- **Decrypt:** If you have an encrypted text, use the decryption box to return to the plain text using a key (K) mentioned before.



3. **DESimage Cipher:** DES is considered a symmetric key block cipher algorithm. The implementation structure of DES is the Feistel cipher. In Feistel structure as 16 rounds of steps are used. 64 bit block size is used for DES structure. It has 64 bit key length, but DES utilizes only 56 bit key. Remaining 8 bits are used later but not used for encryption.

- **Encrypt:** In the encryption method we considered two inputs, one is encryption secret key and another one is original color image. Image file can be reshaped or divided pixel block of original image and express DES encryption process and defining the key for encryption that is secret key. By using DES algorithm procedure finally original image is encrypted with security, this is encrypted image

- **Outputs:** An encrypted image and decrypted image.
- **Decrypt:** It is a reverse process of Image encryption. In this method encrypted image is considered as input for DES algorithm structure for decryption. Encrypted image is divided again into pixel blocks that are same as DES algorithm block length. Primarily function blocks of 64-bit size are entered. Then same secrecy key that is decryption key used for process of decryption which one is used for encryption. Here we follow a reverse ordered procedure of encryption.

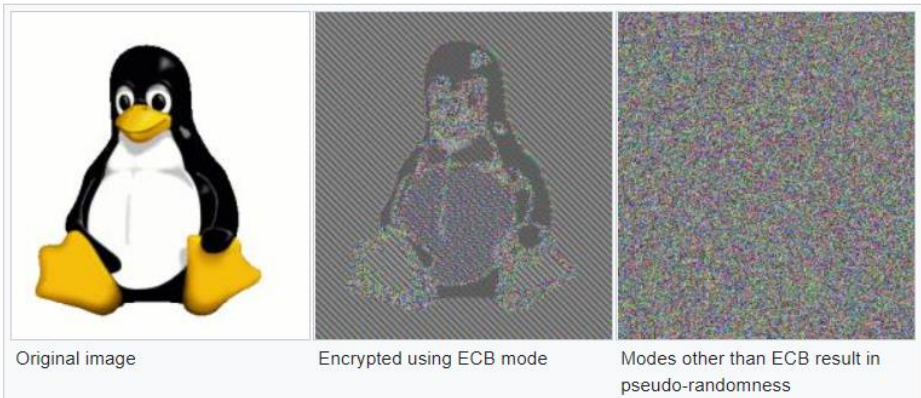
4. AESimage Cipher: The Advanced Encryption Standard (AES) is a symmetric block cipher chosen by the U.S. government to protect classified information. AES is implemented in software and hardware throughout the world to encrypt sensitive data.

Different cipher modes mask patterns by cascading outputs from the cipher block or other globally deterministic variables into the subsequent cipher block. The inputs of the listed modes are summarized in the following table:

We are going to use just five of them: **ECB ,CBC, CFB, OFB, CTR.**

Summary of modes			
Mode		Formulas	Ciphertext
Electronic codebook	(ECB)	$Y_i = F(\text{PlainText}_i, \text{Key})$	Y_i
Cipher block chaining	(CBC)	$Y_i = \text{PlainText}_i \text{ XOR Ciphertext}_{i-1}$	$F(Y, \text{Key}); \text{Ciphertext}_0 = \text{IV}$
Propagating CBC	(PCBC)	$Y_i = \text{PlainText}_i \text{ XOR } (\text{Ciphertext}_{i-1} \text{ XOR PlainText}_{i-1})$	$F(Y, \text{Key}); \text{Ciphertext}_0 = \text{IV}$
Cipher feedback	(CFB)	$Y_i = \text{Ciphertext}_{i-1}$	$\text{Plaintext XOR } F(Y, \text{Key}); \text{Ciphertext}_0 = \text{IV}$
Output feedback	(OFB)	$Y_i = F(Y_{i-1}, \text{Key}); Y_0 = F(\text{IV}, \text{Key})$	$\text{Plaintext XOR } Y_i$
Counter	(CTR)	$Y_i = F(\text{IV} + g(i), \text{Key}); \text{IV} = \text{token}()$	$\text{Plaintext XOR } Y_i$

- Electronic codebook (ECB):** The disadvantage of this method is a lack of diffusion. Because ECB encrypts identical plaintext blocks into identical ciphertext blocks, it does not hide data patterns well. ECB is not recommended for use in cryptographic protocols.

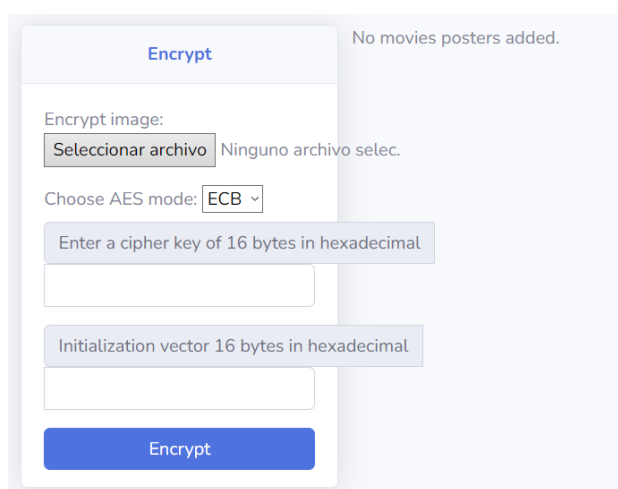


- Cipher block chaining (CBC):** cipher block chaining (CBC) mode of operation in 1976. In CBC mode, each block of plaintext is XORed with the previous ciphertext block before being encrypted. This way, each ciphertext block depends on all plaintext blocks processed up to that point. To make each message unique, an initialization vector must be used in the first block.
- Cipher Feedback (CFB):** The cipher feedback (CFB) mode, in its simplest form uses the entire output of the block cipher. In this variation, it is very similar to CBC, makes a block cipher into a self-synchronizing stream cipher. CFB decryption in this variation is almost identical to CBC encryption performed in reverse.
- Output feedback (OFB):** The output feedback (OFB) mode makes a block cipher into a synchronous stream cipher. It generates keystream blocks, which are then XORed with the plaintext blocks to get the ciphertext. Just as with

CriptoBros – User Manual

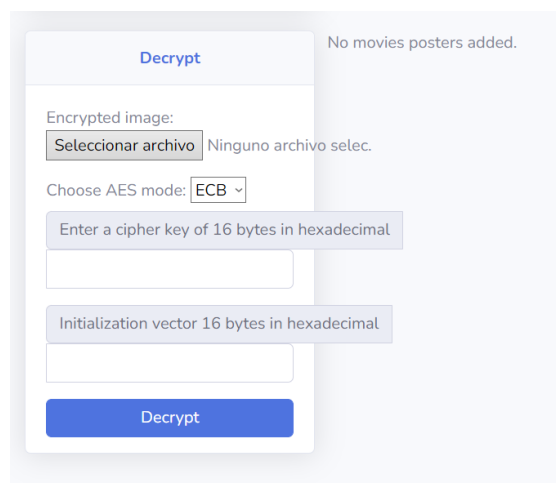
other stream ciphers, flipping a bit in the ciphertext produces a flipped bit in the plaintext at the same location. This property allows many error-correcting codes to function normally even when applied before encryption.

- **Counter (CTR):** CTR mode has similar characteristics to OFB, but also allows a random-access property during decryption. CTR mode is well suited to operate on a multi-processor machine, where blocks can be encrypted in parallel. Furthermore, it does not suffer from the short-cycle problem that can affect OFB.
- **Encrypt:** In the encryption method we considered two inputs, one is encryption secret key and another one is original color image. Image file can be reshaped or divided pixel block of original image and express AES encryption process and defining the key for encryption that is secret key. By using AES algorithm and a mode procedure finally original image is encrypted with security.



The screenshot shows the 'Encrypt' interface. At the top, there is a title 'Encrypt' and a status message 'No movies posters added.' Below the title, there is a section 'Encrypt image:' with a button labeled 'Seleccionar archivo' and the text 'Ninguno archivo selec.' next to it. Below this, there is a section 'Choose AES mode:' with a dropdown menu showing 'ECB'. Below the dropdown, there are two input fields: 'Enter a cipher key of 16 bytes in hexadecimal' and 'Initialization vector 16 bytes in hexadecimal'. At the bottom, there is a blue button labeled 'Encrypt'.

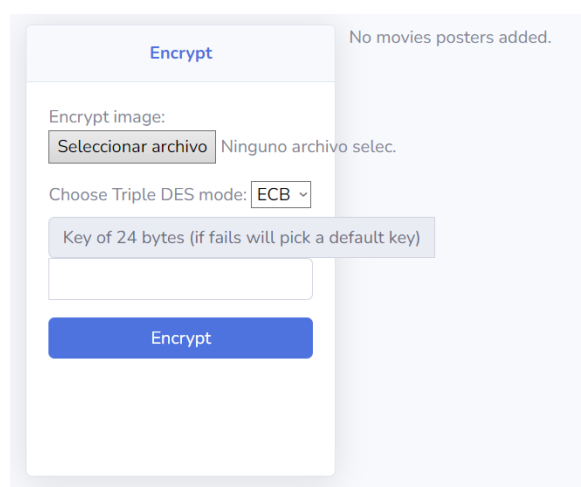
- **Decrypt:** It is a reverse process of Image encryption. In this method encrypted image is considered as input for AES algorithm structure for decryption. Encrypted image is divided again into pixel blocks that are same as AES algorithm block length. Then same secrecy key that is decryption key used for process of decryption which one is used for encryption. Here we follow a reverse ordered procedure of encryption.



The screenshot shows the 'Decrypt' interface. At the top, there is a title 'Decrypt' and a status message 'No movies posters added.' Below the title, there is a section 'Encrypted image:' with a button labeled 'Seleccionar archivo' and the text 'Ninguno archivo selec.' next to it. Below this, there is a section 'Choose AES mode:' with a dropdown menu showing 'ECB'. Below the dropdown, there are two input fields: 'Enter a cipher key of 16 bytes in hexadecimal' and 'Initialization vector 16 bytes in hexadecimal'. At the bottom, there is a blue button labeled 'Decrypt'.

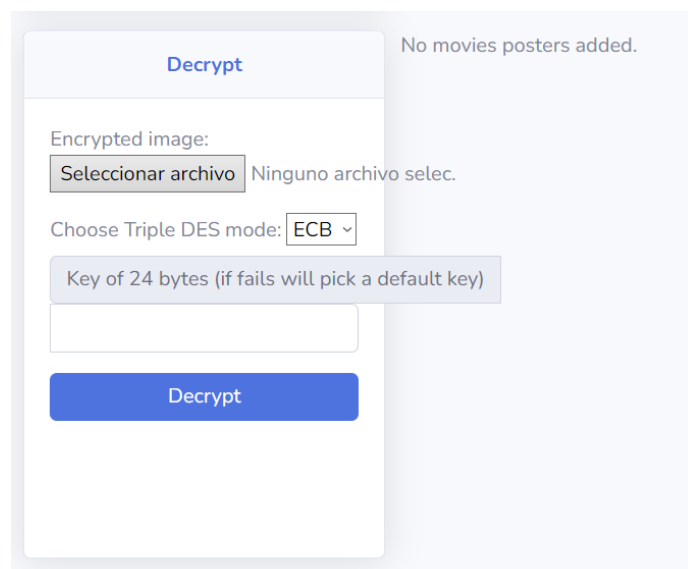
5. TDESImage cipher: It is a symmetric-key block cipher, which applies the DES cipher algorithm three times to each data block. The Data Encryption Standard's (DES) 56-bit key is no longer considered adequate in the face of modern cryptanalytic techniques and supercomputing power. A CVE released in 2016, CVE-2016-2183 disclosed a major security vulnerability in DES and 3DES encryption algorithms. As AESImage cipher we'll be using the same modes.

- **Encrypt:** Triple-DES encryption uses a triple-length DATA key comprised of three 8-byte DES keys to encipher 8 bytes of data using this method: Encipher the data using the first key Decipher the result using the second key Encipher the second result using the third key The procedure is reversed to decipher data that has been triple-DES enciphered: Decipher the data using the third key Encipher the result using the second key Decipher the second result using the first key.



The screenshot shows a web interface for the 'Encrypt' function. At the top, there's a header 'Encrypt' in blue. Below it, a text label 'Encrypt image:' is followed by a file selection button labeled 'Seleccionar archivo' and a status message 'Ninguno archivo selec.'. Below this, there's a label 'Choose Triple DES mode:' followed by a dropdown menu showing 'ECB'. Underneath is a text input field for the 'Key of 24 bytes (if fails will pick a default key)'. At the bottom is a large blue button labeled 'Encrypt'. The background is a light blue gradient with a faint image of a movie poster.

- **Decrypt:** It is a reverse process of Image encryption. In this method encrypted image is considered as input for 3DES algorithm structure for decryption. Encrypted image is divided again into pixel blocks that are same as 3DES algorithm block length. Then same secrecy key that is decryption key used for process of decryption which one is used for encryption. Here we follow a reverse ordered procedure of encryption.



The screenshot shows a web interface for the 'Decrypt' function. At the top, there's a header 'Decrypt' in blue. Below it, a text label 'Encrypted image:' is followed by a file selection button labeled 'Seleccionar archivo' and a status message 'Ninguno archivo selec.'. Below this, there's a label 'Choose Triple DES mode:' followed by a dropdown menu showing 'ECB'. Underneath is a text input field for the 'Key of 24 bytes (if fails will pick a default key)'. At the bottom is a large blue button labeled 'Decrypt'. The background is a light blue gradient with a faint image of a movie poster.

For a given cryptographic system, the minimum length of a ciphertext to be able to be cryptanalyzed can be found using $\frac{\log_2 |K|}{\log_2 |P| - H_L}$, where $|K|$ the size of our key space, $|P|$ the size of our alphabet, and H_L the entropy of a given language. In this case, we want to do the analysis for the Vigenère cipher, for the English language, so our $|P| = 26$, $H_L \approx 1.5$ and our $|K| = 26^n$, for a given key length n . Then:

$$\begin{aligned} \frac{\log_2 |K|}{\log_2 |P| - H_L} &= \frac{\log_2 26^n}{\log_2 26 - 1,5} \\ &= \frac{n \log_2 26}{\log_2 26 - 1,5} \\ &= \frac{n * 4,7}{4,7 - 1,5} \\ &= \frac{n * 4,7}{3,2} \\ &= \frac{n * 4,7}{3,2} \\ &\approx n * 1,47 \end{aligned}$$

Therefore, for a key of length n , we need a ciphertext $1.47 * n \approx 1.5 * n$. For example, for a key of length $n = 3$, a ciphertext of minimum length $n = 1.5 * 3 = 4.5 \approx 5$ would be needed, or for a key of length $n = 6$, then length 9. However, this is just a theoretical limit, for an opponent with unlimited resources. In general, and as could be seen with our tests, a larger key is required to properly perform cryptanalysis.

6. Gamma-Pentagonal Cipher: The Gamma-Pentagonal cryptographic system has the property of probabilistic security since it is associated with a difficult problem of number theory. More precisely, the problem we are dealing with consists in determining the number of ways in which an integer can be written as the sum of three polygonal numbers. It is recalled that polygonal numbers are those that can be described as a polygonal arrangement of points, for example, triangular numbers, squares, pentagonal, etc.

- **Encrypt:** The difficult problem is to determine in how many ways a positive integer can be written as a sum of at least three square numbers. The above problem can be seen as finding how many admissible trajectories such as the ones we will illustrate later, connect the origin with a point in the usual plane. Then the point is related to a letter. The set-up process is as follows:

- The user must provide the `origin` coordinates, a `permutation` and the `clear text`. The origin and the permutation must remain secret.
- A matrix of 10 columns is created, each of which contains the complete alphabet (26 letters). Then the order of each column is shifted, the magnitude of the displacement is determined by the respective number in the initial permutation.
- At the same time, the following equivalence classes are created:

$$\bar{x} := [(a, b) : a + b = x]$$

Then, $d(\bar{x})$ is defined as the sum of the incoming arrows for each point in the class.

- Finally, the value of each $a_{i,j}$ in the matrix is shifted by $d((i, j))$ units.

Encrypt

Text

Permutation

Start

Encrypt

- **Decrypt:** The user must provide the origin coordinates, the permutation and the ciphered coordinates. After doing the set-up, the coordinates are translated into letters, returning the original clear text. Since the keys in each alphabet can be used with the same probability when constructing ciphertexts then we can infer that the gamma-pentagonal system is unbreakable.

Decrypt

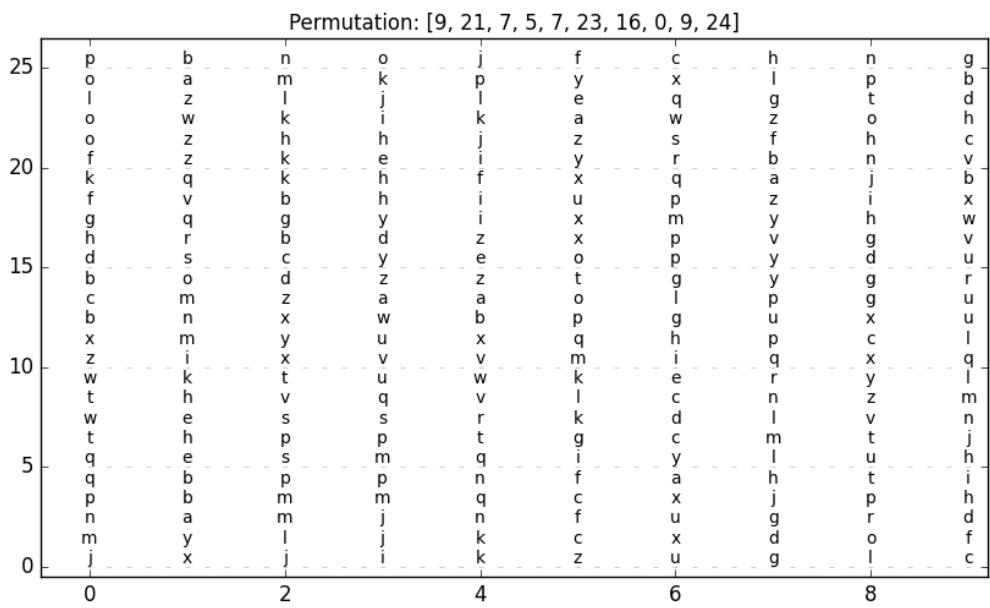
Ciphered coordinates

Permutation

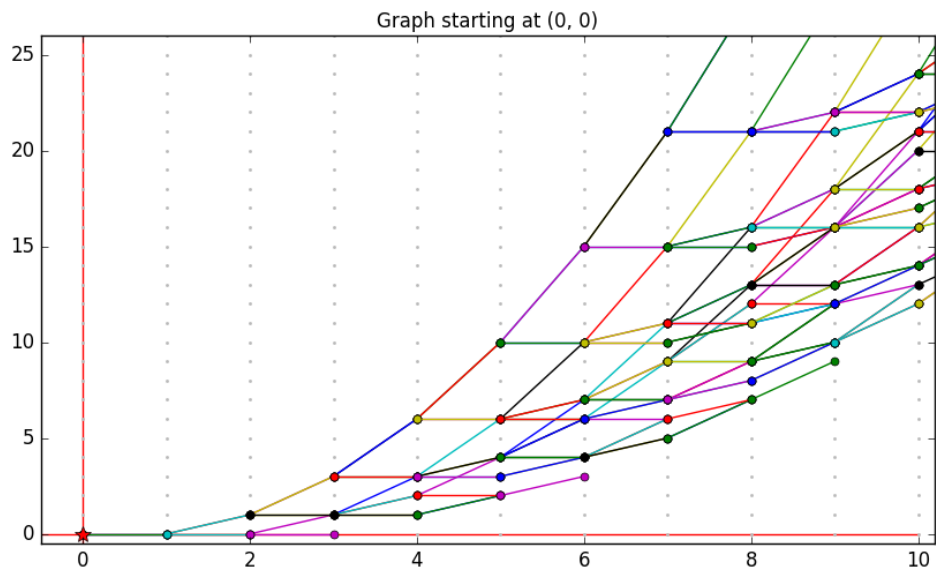
Start

Decrypt

As an example:



A graph generated with (0,0) as the origin point is shown below:



6. Public-key Cryptosystems

In a public-key encryption system, anyone with a public key can encrypt a message, yielding a ciphertext, but only those who know the corresponding private key can decrypt the ciphertext to obtain the original message. In a digital signature system, a sender can use a private key together with a message to create a signature. Anyone with the corresponding public key can verify whether the signature matches the message, but a forger who doesn't know the private key can't find any message/signature pair that will pass verification with the public key.

1. **RSA Cryptosystem:** The algorithm can be used for both confidentiality (encryption) and authentication (digital signature). It is worth noting that signing and decryption are significantly slower than verification and encryption. The cryptographic strength is primarily linked to the length of the RSA modulus n .

```
Crypto.PublicKey.RSA.import_key(extern_key, passphrase=None)
```

Import an RSA key (public or private).

- Parameters:**
- **extern_key** (*string or byte string*) – The RSA key to import.
The following formats are supported for an RSA **public** key:
 - X.509 certificate (binary or PEM format)
 - X.509 `subjectPublicKeyInfo` DER SEQUENCE (binary or PEM encoding)
 - PKCS#1 `RSAPublicKey` DER SEQUENCE (binary or PEM encoding)
 - An OpenSSH line (e.g. the content of `~/.ssh/id_ecdsa`, ASCII)
 The following formats are supported for an RSA **private** key:
 - PKCS#1 `RSAPrivateKey` DER SEQUENCE (binary or PEM encoding)
 - PKCS#8 `PrivateKeyInfo` or `EncryptedPrivateKeyInfo` DER SEQUENCE (binary or PEM encoding)
 - OpenSSH (text format, introduced in [OpenSSH 6.5](#))
 For details about the PEM encoding, see [RFC1421/RFC1423](#).
 - **passphrase** (*string or byte string*) – For private keys only, the pass phrase that encrypts the key.

Returns: An RSA key object (`RsaKey`).

Raises: `ValueError/IndexError/TypeError` – When the given key cannot be parsed (possibly because

- **Encrypt:** In the box shown below, we write the encrypted message we want to decrypt, then by pressing submit we get a result we will be downloading, giving us an output file with decrypted code. The messages sent are represented by numbers, and the operation is based on the known product of two large prime numbers chosen at random and kept secret. Currently these primes are of the order of 10^{300} , and it is expected that their size will continue to grow as the computing power of computers increases.

RSA Cipher

Encrypt

Submit

Outputs

- **Decrypt:** We first separate the message from encrypted message digest, we then decrypt the message digest using sender public key $S' = M^e \bmod n$. We then calculate the message digest of the message using SHA-224, M' . If $M' = S'$, then the sender is indeed the real sender and the message is not been tampered with.

Decrypt

8-bit text in the following format:10001001 (with out spaces)

10-bit key in the following format:1000100111 (with out spaces)

Submit

2. **The Rabin Cryptosystem:** The Rabin cryptosystem is a family of public-key encryption schemes based on a trapdoor function whose security, like that of RSA, is related to the difficulty of integer factorization. Like all asymmetric cryptosystems, the Rabin system uses a key pair: a public key for encryption and a private key for decryption. The public key is published for anyone to use, while the private key remains known only to the recipient of the message.

Algorithm 10.6 *Key generation for Rabin cryptosystem*

```
Rabin_Key_Generation
{
    Choose two large primes  $p$  and  $q$  in the form  $4k + 3$  and  $p \neq q$ .
     $n \leftarrow p \times q$ 
    Public_key  $\leftarrow n$  // To be announced publicly
    Private_key  $\leftarrow (q, n)$  // To be kept secret
    return Public_key and Private_key
}
```

- **Encrypt:** Get the public key n . Convert the message to ASCII value. Then convert it to binary and extend the binary value with itself, and change the binary value back to decimal m . Encrypt with the formula: $C = m^2 \bmod n$, Send C to the recipient.
 - **Decrypt:** Accept C from sender, then specify a and b with Extended Euclidean GCD such that, $a.p + b.q = 1$, Compute r and s using following formula: $r = C^{(p+1)/4} \bmod p$, $s = C^{(q+1)/4} \bmod q$. Now, calculate X and Y using following formula: $X = (a.p.r + b.q.s) \bmod p$, $Y = (a.p.r - b.q.s) \bmod q$. The four roots are, $m1=X$, $m2=-X$, $m3=Y$, $m4=-Y$. Now, Convert them to binary and divide them all in half. Determine in which the left and right half are same. Keep that binary's one half and convert it to decimal m . Get the ASCII character for the decimal value m . The resultant character gives the correct message sent by sender.
3. **ElGamal Cryptosystem:** The ElGamal signature scheme is a digital signature scheme based on the algebraic properties of modular exponentiation, together with the discrete logarithm problem. The algorithm uses a key pair consisting of a public key and a private key. The private key is used to generate a digital signature for a message, and such a signature can be verified by using the signer's corresponding public key. The digital signature provides message authentication (the receiver can verify the origin of the message), integrity (the receiver can verify that the message has not been modified since it was signed) and non-repudiation (the sender cannot falsely claim that they have not signed the message).
- **ElGamal cipher:** The Elgamal public key encryption algorithm is based on the Discrete logarithm problem (DLP) and is closely related to Diffie-Hellman key exchange method. In this page we implement the version of the Elgamal PKC that is based on the discrete logarithm problem for Fields, but the construction works quite generally using the DLP in any group. In particular, we implemented a version of the Elgamal PKC based on elliptic curve groups in another page.
 - **Encrypt:** key generator: Let us suppose there are n users and a particular user u_i randomly select a secret key $d_i \in \mathbb{Z}_p^*$, where $i \in 1, 2, 3 \dots n$. The public key corresponding to user u_i is $y_i = g^{d_i} \bmod p$. Public key = g, p, y_i and private key = d_i .

The screenshot shows a web interface titled "Encryption Keys". It contains two text input fields. The first field is labeled "Public Key" and has a small downward-pointing arrow on its right side, indicating a dropdown menu. The second field is labeled "Private Key". Below both input fields is a prominent blue button with the text "Generate Encryption Keys" in white.

Encrypt

Completa este campo

Public Key

Submit

ElGamal PKC same parameter r has to repeat the number of times of message, hence system is vulnerable to known-plaintext attack. We are proposing variance of ElGamal cryptosystem so that large messages can be encrypted using the existing technology with little modification on it. As we know traditional ElGamal cryptosystem having two parameters r and d on which security of system relay, as these two are randomly selected and kept secret, therefore we are extending these two parameters in order to encrypt more numbers of messages.

- **Decrypt:** The receiver uses its private key (d_1, d_2) to obtain plain texts. The plaintexts messages obtain using cipher-text are:

$$M_1 = C_1 * (b^{d_1})^{-1} \bmod p.$$

$$M_2 = C_2 * (b^{d_2})^{-1} \bmod p.$$

$$M_3 = C_3 * (b^{d_1 + d_2})^{-1} \bmod p.$$

Decrypt

Completa este campo

Private Key

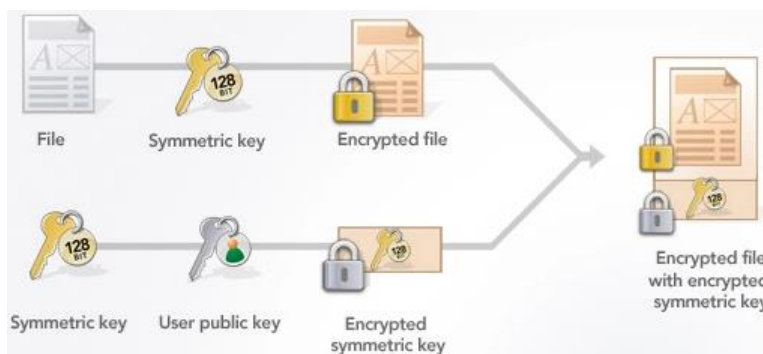
Submit

Outputs

Public Key used to encrypt

Private Key used to decrypt

- **Elptic Curve Criptography:** Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC allows smaller keys compared to non-EC cryptography (based on plain Galois fields) to provide equivalent security. Elliptic curves are applicable for key agreement, digital signatures, pseudo-random generators and other tasks. Indirectly, they can be used for encryption by combining the key agreement with a symmetric encryption scheme.
 - **Encrypt:** The elliptic curve cryptography (ECC) does not directly provide encryption method. Instead, we can design a hybrid encryption scheme by using the ECDH (Elliptic Curve Diffie–Hellman) key exchange scheme to derive a shared secret key for symmetric data encryption and decryption.



Encrypt

Public Key

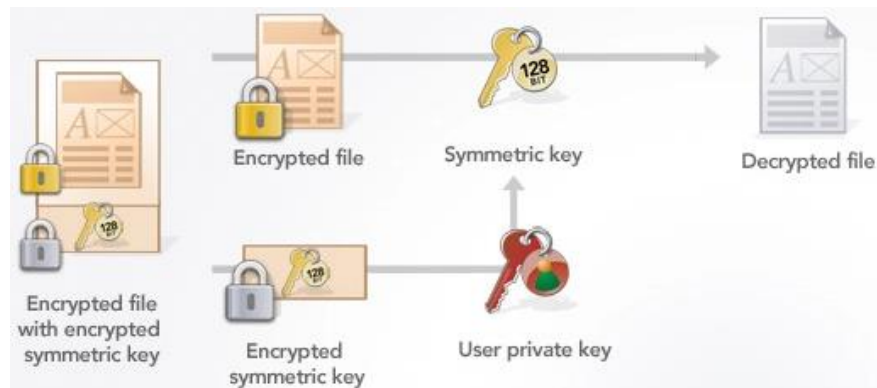
Submit

Outputs

Public Key used to encrypt

Private Key used to decrypt

- **Decrypt:** To decrypt the encrypted message, we use the data produced during the encryption, along with the decryption privateKey. The result is the decrypted plaintext message. We use authenticated encryption (GCM block mode), so if the decryption key or some other parameter is incorrect, the decryption will fail with an exception.



The screenshot shows a web interface titled 'Decrypt'. It contains a large, empty text input box. To the right of the bottom of this box is a small black box with the word 'Complete' in white. Below the text box is a 'Private Key' label followed by an input field. At the bottom of the interface is a blue 'Submit' button.

- **Digital signature Authenticator:** The Elliptic Curve Digital Signature Algorithm (ECDSA) is a Digital Signature Algorithm (DSA) which uses keys derived from elliptic curve cryptography (ECC). It is a particularly efficient equation based on public key cryptography (PKC). ECDSA is used across many security systems, is popular for use in secure messaging apps, and it is the basis of Bitcoin security (with Bitcoin "addresses" serving as public keys).
 - **Encryption Keys:** As with elliptic-curve cryptography in general, the bit size of the private key believed to be needed for ECDSA is about twice the size of the security level, in bits. For example, at a security level of 80 bits—meaning an attacker requires a maximum of about 2^{80} operations to find the private key—the size of an ECDSA private key would be 160 bits.
 - **Encrypt:** Initially, they must agree on the curve parameters. In addition to the field and equation of the curve, we need G , a base point of prime order on the curve; n is the multiplicative order of the point G .

Encrypt

Private Key

Submit

Encryption Keys

Veri
ficat
ion
Key

Priv
ate
Key

Generate Encryption Keys

- **Authenticate :** The signature verification works if a signature on a message was indeed generated by a verification Key.

Authenticate

Verification Key

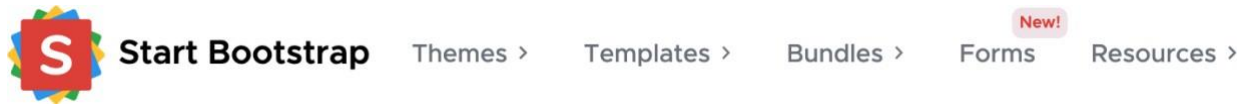
Original Text

Submit

Outputs

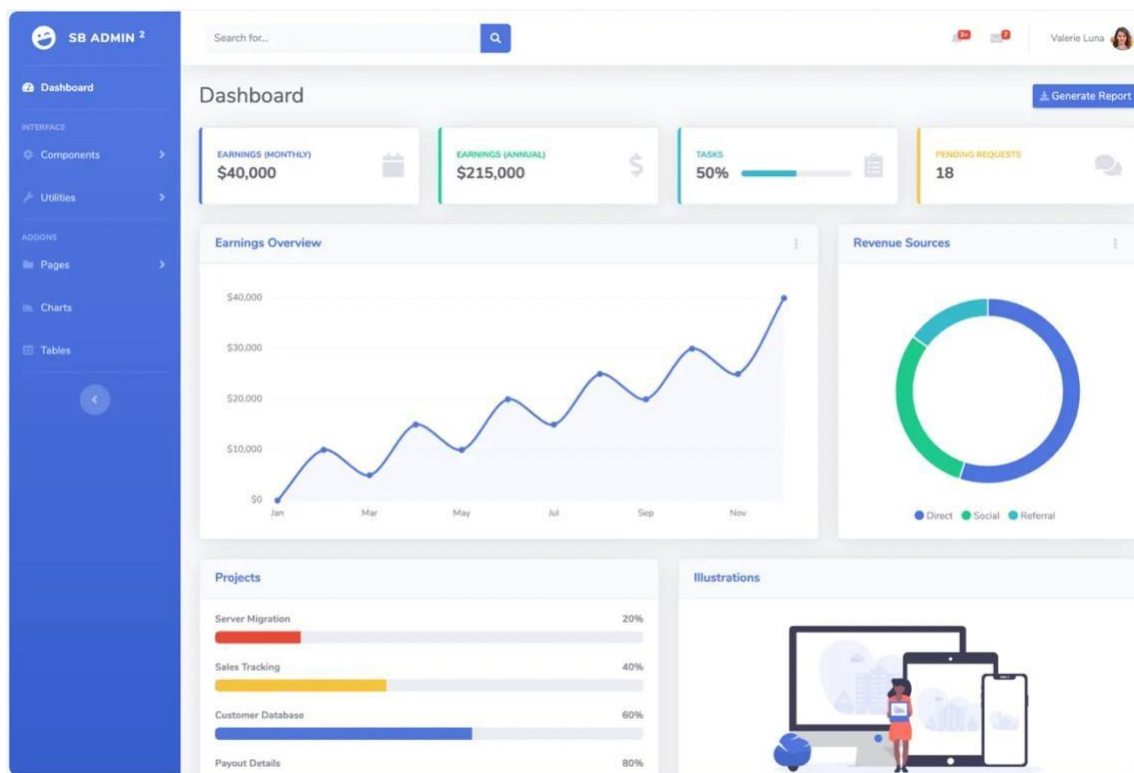
7. Page Design:

The page Design was inspired by a free open-source start Bootstrap theme.

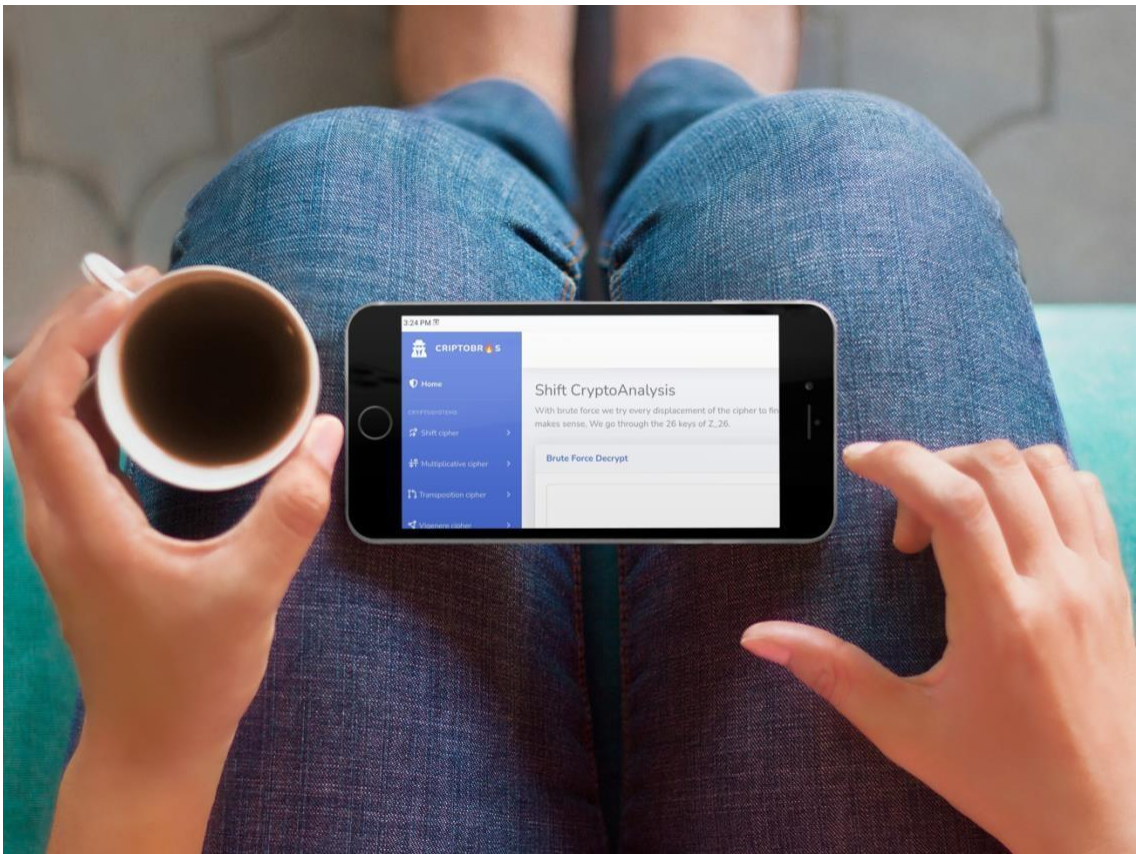


- Template:

We used a template called SB Admin 2 is a free, open-source, Bootstrap 4-based admin theme perfect for quickly creating dashboards and web applications. Its modern design style with subtle shadows and a card-based layout could be described as flat material and is inspired by the principles of material design along with a simple, attractive color system.



We got some features like Features: A modern, material design-inspired layout, a focus on utility, classes to minimize CSS bloat, custom card, and button components, and custom utility classes forextended functionality.



•Color:

There are many ways to create a website that stands out on the Internet, and one of them is to choose a unique color scheme. Whether you're designing a blog, an online store, or a personal page, the color selection for a website is one of the first things visitors will notice, and you're sure to make a lasting impression. We used kind a color bust:

Using a gradient background on your web page can set the tone for a wide-gamut color palette. In the case of Foodie Marketing, a pop of pink and orange hues inspires a cool contrast of teal, blue, and lime green. The white logo, text, and buttons add a professional touch to the page's vibrant vibe.

<div>Primary</div> <div>#4e73df</div>	<div>Success</div> <div>#1cc88a</div>
<div>Info</div> <div>#36b9cc</div>	<div>Warning</div> <div>#f6c23e</div>
<div>Danger</div> <div>#e74c3b</div>	<div>Secondary</div> <div>#858796</div>
<div>Light</div> <div>#f8f9fc</div>	<div>Dark</div> <div>#5a5c69</div>