1. Key - The key is a secret value that is used to encrypt and decrypt data. It must be a multiple of 128 bits (16 bytes) or 256 bits (32 bytes) in length, depending on the AES variant used. The key is usually generated randomly or derived from a password using a key derivation function.
2. Plaintext - The plaintext is the data that is to be encrypted. It can be any data in binary format, such as a file or a message.
3. Initialization Vector (IV) - The IV is a random value that is used to ensure that the same plaintext will produce different ciphertext every time it is encrypted with the same key. The IV is a 16-byte value for AES-128 and a 32-byte value for AES-256.

Optional parameters for AES encryption include:

1. Mode of Operation - AES can be used in various modes of operation to provide additional security, such as CBC (Cipher Block Chaining), CTR (Counter), or GCM (Galois/Counter Mode).
2. Padding - AES works on blocks of 128 bits, so if the plaintext is not a multiple of 128 bits, it must be padded to ensure that it can be encrypted. There are several padding schemes, such as PKCS#7 or ISO/IEC 7816-4.
3. Key Expansion - In order to generate the key schedule, the key is expanded into a larger set of round keys. The number of rounds depends on the key size and the block size, and it can be 10, 12, or 14 for AES-128, AES-192, and AES-256, respectively.
4. Salt - In some cases, a salt value may be used to add randomness to the key derivation process, making it more resistant to attacks.

Rot 47

ROT13 (rotate by 13 places) is a simple Caesar cipher technique that replaces each letter of the alphabet with the letter 13 places ahead of it in the alphabet. There is only one parameter required for ROT13 encryption:

1. Plaintext - The plaintext is the data that is to be encrypted. It can be any text in ASCII format, such as a message or a file.

To encrypt the plaintext using ROT13, simply replace each letter in the text with the letter 13 places ahead of it in the alphabet. For example, the letter "A" would be replaced with the letter "N", "B" with "O", and so on. When you reach the end of the alphabet, start again from the beginning (i.e. "Z" becomes "M", "Y" becomes "L", and so on).

It's important to note that ROT13 is a very weak encryption method and is not suitable for secure communication or data protection. It is primarily used for obfuscation or as a simple way to encode text in situations where a small amount of security is required.

Ui design

Inputs:

Password

File

Encryption type.

Outputs:

Password?

Plain text encrypted text.

Visual:

Dark mode “Hacker mode?”

Loading bar

File -> AES + XOR

Password -> Rot47

DES:

DES (Data Encryption Standard) is a symmetric-key encryption algorithm used for securing electronic data. It was developed by IBM in the 1970s and later adopted by the U.S. government as a standard for protecting sensitive data.

In DES, the same secret key is used for both encryption and decryption of the data. The key length is 56 bits, which means that there are 2^56 possible keys. The algorithm uses a series of substitution and permutation operations to encrypt the data.

The DES algorithm works in several stages:

1. Key generation: The 56-bit key is expanded to 64 bits and then divided into 16 48-bit subkeys.
2. Initial permutation: The 64-bit input block is permuted according to a fixed table.
3. Feistel rounds: The input block is divided into two 32-bit halves, and a series of 16 rounds are performed on these halves using the subkeys generated in step 1. Each round involves a combination of substitution and permutation operations.
4. Final permutation: After the 16 rounds are completed, the two 32-bit halves are combined and permuted again according to a fixed table.

The result of this process is the encrypted data, which can be decrypted using the same key and the reverse of the DES algorithm. While DES was once widely used, it is now considered insecure due to its relatively short key length and vulnerability to brute force attacks. It has largely been replaced by newer and more secure encryption algorithms like AES (Advanced Encryption Standard).

Sure, here are the full list of steps of the DES algorithm:

1. Key Generation:

* The 56-bit secret key is permuted using a fixed table called PC-1 (Permuted Choice-1) to produce a 56-bit key with parity bits.
* The 56-bit key is then divided into two 28-bit halves.
* Each half is subjected to a circular shift operation, either one or two bits to the left, depending on the round of encryption, to create a new pair of 28-bit halves.
* The new 56-bit key is produced by combining the two shifted halves and permuting the result using a fixed table called PC-2 (Permuted Choice-2) to produce 16 48-bit subkeys, one for each round of encryption.

1. Initial Permutation:

* The 64-bit plaintext block is permuted using a fixed table called IP (Initial Permutation).

1. Feistel Rounds:

* The 64-bit plaintext block is divided into two 32-bit halves, left and right.
* For each of the 16 rounds, the right half is expanded from 32 bits to 48 bits using a fixed table called E (Expansion).
* The resulting 48-bit block is XORed with a 48-bit subkey.
* The XOR result is divided into eight 6-bit blocks, which are substituted using eight S-boxes.
* The outputs of the S-boxes are combined into a 32-bit block using a fixed table called P (Permutation).
* The 32-bit block is then XORed with the left half of the plaintext block to produce the new right half.
* The left and right halves are swapped before the next round of encryption.

1. Final Permutation:

* After the 16 rounds of encryption are completed, the left and right halves of the encrypted block are combined and permuted using a fixed table called IP^-1 (Inverse Initial Permutation) to produce the final 64-bit encrypted block.

The result of this process is the encrypted data, which can be decrypted using the same key and the reverse of the DES algorithm.

Logical

Key input text

Key -> Rot 47 Text

Rot 47 -> hash

* 16 byte
* 8 byte

AED encrypt :

Hash 16 byte

Iv is created.

Input file

Output fase1.bin

Convert fase1.bin -> fase2.bin

Des encrypt :

Hash 8 byte

Use iv

Input fase2.bin

Output cc-file