Understanding Permutation Cipher and How to Implement with Python

Aug 15, 2023 • 4 min read
Tags: Cryptography, Python

A permutation cipher is a type of encryption that rearranges the letters of a plaintext message according to a predetermined pattern. This pattern is called the "key". Unlike substitution ciphers, which replace characters with other characters, permutation ciphers maintain the original characters but change their order.

How permutation cipher works?

To encrypt a message using permutation cipher, you first need the key. The key can be a pattern of letters or numbers. The message then divided into several blocks. For each block, we rearrange the position for each character based on the key.

Here is an example:

1. We want to encrypt the message `THE HACKING BLOG ROCKS` and the key is 31254.

2. The length of the key is 5, so divide the message into blocks of 5 characters: `THE H`, `ACKIN`, `G BLO`, `G ROC`, `KS  `. Note that the last block has only 2 character, so we pad it with 3 spaces to make the block contains 5 characters.

3. For each block, we need to rearrange the position of the characters. The key is 31254, so we put the 3rd char into the 1st position, the 1st char to the 2nd position, the 2nd char to the 3rd position, the 5th char to the 4th position, and the 4th char to the 5th position.

4. The encrypted message is `ETHH KACNIBG OLRG CO KS  `.

To decrypt the message, we use the same algorithm but reversing the character placement:

1. We want to decrypt the message `ETHH KACNIBG OLRG CO KS  ` and the key is 31254.

2. We divide the message into blocks of 5 characters: `ETHH `, `KACNI`, `BG OL`, `RG CO`, ` KS  `.

3. For each block, rearrange the position of the characters. The key is 31254, so we put the 1st char into the 3rd position, the 2nd char to the 1st position, the 3rd char into the 2nd position, the 4th char into the 5th position, and the 5th char into the 4th position.

4. he decrypted message is `THE HACKING BLOG ROCKS  `.

How do I implement it in Python?

The algorithm for encryption and decryption is similar, with the difference only in the character placement.

Below is a function to encrypt and decrypt text with permutation cipher.

```
def permutation_cipher(msg: str, key: int, decrypt: bool=False) -> str:
    key = str(key)
    key_len = len(key)
    result = ''

    # Pad the message with spaces if needed
    msg += ' ' * ((key_len - (len(msg) % key_len)) % key_len)

    # Calculate the number of blocks
    num_blocks = int(len(msg) / key_len)

    # Process each block
    for num_block in range(num_blocks):
        # Get the current block
        start_idx = num_block * key_len
        block = [x for x in msg[start_idx:(start_idx + key_len)]]

        # Rearrange the characters from this block into a new block
        new_block = [' '] * key_len
        for idx1, idx2 in enumerate(key):
            idx2 = int(idx2) - 1
            if decrypt:
                new_block[idx2] = block[idx1]
            else:
                new_block[idx1] = block[idx2]

        # Join the new block into the result message
        result += ''.join(new_block)

    return result
```

```
# Usage example
key = 31254
message = 'THE HACKING BLOG ROCKS'

encrypted = permutation_cipher(message, key)
print(encrypted)   # Should print "ETHH KACNIBG OLRG CO KS  "

decrypted = permutation_cipher(encrypted, key, decrypt=True)
print(decrypted)   # Should print "THE HACKING BLOG ROCKS   "
```

The function `permutation_cipher` takes 3 arguments: `msg` is the message to be encrypted or decrypted, `key` is the encryption key, and `decrypt` to specify whether the operation is encryption or decryption.

First, the input message is padded with spaces to make the blocks contain equal number of characters. Then for each block, a new block is created and filled with the input characters based on the given key. The new blocks then joined together and returned as output.

Key Takeaways

In this blog post, we have learned how the permutation cipher works and how to implement it in Python. Permutation ciphers are a simple and effective way to encrypt short messages.

Keep in mind that they should not be used for encrypting sensitive information. If you are looking for a more secure way to encrypt your messages, you should look into the more modern and secure algorithms such as RSA and AES.