Playfair Cipher: Theoretical Overview

The Playfair cipher, invented by Charles Wheatstone in 1854 and promoted by Lord Playfair, is a digraph substitution cipher. It encrypts pairs of letters, offering significant advantages over simple monoalphabetic substitution ciphers.

Features

- Uses a 5x5 grid of letters based on a keyword
- Encrypts digraphs (pairs of letters) instead of single letters
- 25 letters (I and J are typically combined)
- Resistant to frequency analysis attacks

Key Generation

1. Choose a keyword and remove duplicate letters
2. Fill a 5x5 matrix with the keyword letters first
3. Complete the matrix with remaining alphabet letters
4. Example with keyword "MONARCHY":

$$\begin{array}{ccccc} M & O & N & A & R \\ C & H & Y & B & D \\ E & F & G & I/J & K \\ L & P & Q & S & T \\ U & V & W & X & Z \end{array}$$

Encryption Rules

For each pair of plaintext letters (a, b):

1. If a and b are in the same row, replace with letters to their right (wrapping around)
2. If a and b are in the same column, replace with letters below (wrapping around)
3. Otherwise, replace with letters on the same row but in the column of the other letter

Security Considerations

- Stronger than simple substitution ciphers
- Vulnerable to known-plaintext attacks
- Frequency analysis of digraphs can be used for cryptanalysis
- Modern cryptography has rendered it obsolete for secure communication

Listing 1: Playfair Cipher Implementation

```
def prepare_key(key):
    key = key.upper().replace("J", "I")
    key = "".join(dict.fromkeys(key + "ABCDEFGHIKLMNOPQRSTUVWXYZ"))
    matrix = [list(key[i:i+5]) for i in range(0, 25, 5)]
    return matrix

def find_position(matrix, char):
    for i, row in enumerate(matrix):
        if char in row:
            return i, row.index(char)

def playfair_encrypt(plaintext, key):
    matrix = prepare_key(key)
    plaintext = plaintext.upper().replace("J", "I")
    plaintext = [plaintext[i:i+2] for i in range(0, len(plaintext), 2)]

    if len(plaintext[-1]) == 1:
        plaintext[-1] += 'X'

    ciphertext = ""
    for pair in plaintext:
        r1, c1 = find_position(matrix, pair[0])
        r2, c2 = find_position(matrix, pair[1])

        if r1 == r2:
            ciphertext += matrix[r1][(c1+1)%5] + matrix[r2][(c2+1)%5]
        elif c1 == c2:
            ciphertext += matrix[(r1+1)%5][c1] + matrix[(r2+1)%5][c2]
        else:
            ciphertext += matrix[r1][c2] + matrix[r2][c1]

    return ciphertext

def playfair_decrypt(ciphertext, key):
    matrix = prepare_key(key)
    ciphertext = [ciphertext[i:i+2] for i in range(0, len(ciphertext), 2)]

    plaintext = ""
    for pair in ciphertext:
        r1, c1 = find_position(matrix, pair[0])
        r2, c2 = find_position(matrix, pair[1])

        if r1 == r2:
            plaintext += matrix[r1][(c1-1)%5] + matrix[r2][(c2-1)%5]
        elif c1 == c2:
            plaintext += matrix[(r1-1)%5][c1] + matrix[(r2-1)%5][c2]
        else:
            plaintext += matrix[r1][c2] + matrix[r2][c1]

    return plaintext

# Example usage
key = "KEYWORD"
plaintext = "HELLOWORLD"
ciphertext = playfair_encrypt(plaintext, key)
decrypted = playfair_decrypt(ciphertext, key)

```

```
58
59
60 print(f"Plaintext: {plaintext}")
61 print(f"Ciphertext: {ciphertext}")
62 print("Matrix:")
63 (__import__("pprint").pprint(prepare_key(key)))
64 print(f"Decrypted: {decrypted}")
```