Laporan Matematika Diskrit

Kelompok 3

Anggota:

Aaron Jevon Benedict Kongdoh
0806022310014

Felicia Wijaya
0806022310007

Javin Erasmus Clementino
0806022310025

Encryption and Decryption Using XOR and SHA-256 Key Generation

Proyek ini mengimplementasikan sistem enkripsi dan dekripsi sederhana namun efektif menggunakan operasi XOR, dengan kunci-kunci yang dihasilkan dari kunci rahasia melalui fungsi hash SHA-256. Sistem ini mengenkripsi plaintext dengan menerapkan serangkaian operasi XOR menggunakan empat kunci biner (B1, B2, B3, B4) yang diperoleh dari kunci rahasia, dan proses yang sama digunakan untuk dekripsi.

1. Key Generation (`generate_keys`)

Fungsi `generate_keys` menghasilkan empat kunci biner dari kunci rahasia dengan melakukan hashing menggunakan algoritma SHA-256. SHA-256 menghasilkan hash sepanjang 256-bit, dan empat byte pertama dari hash ini digunakan sebagai kunci (B1, B2, B3, B4).

Kode:

```
def generate_keys(secret_key):

    hash_key = hashlib.sha256(secret_key.encode()).digest()

    B1 = hash_key[0]

    B2 = hash_key[1]

    B3 = hash_key[2]

    B4 = hash_key[3]

    return B1, B2, B3, B4
```

Sebagai contoh, dengan kunci rahasia "maka", hash SHA-256 menghasilkan array byte, dan empat byte pertama digunakan sebagai kunci.

2. Encryption and Decryption (`xor_encrypt_decrypt`)

Proses enkripsi dan dekripsi dilakukan dengan menggunakan rangkaian operasi XOR dengan empat kunci biner yang dihasilkan dari kunci rahasia. Proses ini sama untuk enkripsi dan dekripsi karena operasi XOR bersifat reversibel.

Proses Enkripsi:

```
def xor_encrypt_decrypt(value, keys):
    result = value
    for key in keys:
        result ^= key
    return result
```

Langkah-langkah:

1. Ubah string plaintext menjadi representasi bilangan bulat.
2. Terapkan operasi XOR secara berurutan dengan kunci B1, B2, B3, dan B4 untuk menghasilkan ciphertext.
3. Untuk dekripsi, proses yang sama diterapkan pada ciphertext dengan kunci yang sama, yang mengembalikan plaintext asli.

3. Example Usage

Misalkan plaintext adalah `"halo"` dan kunci rahasia adalah `"maka"`.

a. Konversi Plaintext ke Bentuk Bilangan Bulat

String `"halo"` dikonversi menjadi bentuk bilangan bulat untuk operasi XOR:

```
plaintext = int.from_bytes(plaintext_str.encode(), 'big')
```

Representasi bilangan bulat dari `"halo"` adalah:

```
plaintext = 1751606880
```

b. Key Generation

Kunci rahasia `"maka"` menghasilkan kunci-kunci berikut setelah menjalankan fungsi `generate_keys`:

```
B1 = 66, B2 = 107, B3 = 157, B4 = 184
```

c. Encryption Process

Untuk mengenkripsi plaintext:

```
ciphertext = xor_encrypt_decrypt(plaintext, keys)
```

Ini menghasilkan ciphertext seperti:

```
ciphertext = 1751606672
```

d. Decryption Process

Untuk mendekripsi ciphertext, operasi XOR yang sama diterapkan:

```
decrypted_text = xor_encrypt_decrypt(ciphertext, keys)
```

Bilangan bulat hasil dekripsi dikonversi kembali ke bentuk string aslinya, yang sesuai dengan plaintext awal:

```
decrypted_str = "halo"
```

4. Results

- Secret Key: `"maka"`
- Plaintext (A): `"halo"`
- Generated Keys (B1, B2, B3, B4): 66, 107, 157, 184
- Ciphertext (C): 1751606672
- Decrypted Text: `"halo"`

5. Conclusion

Metode enkripsi dan dekripsi berbasis XOR yang digunakan di sini sederhana dan efisien. Dengan memperoleh kunci dari hash SHA-256 sebuah kunci rahasia, kita meningkatkan keamanan sistem ini, sehingga sulit untuk menebak atau merekayasa balik kunci-kunci tersebut. Sistem ini efektif dalam mengenkripsi dan mendekripsi data dengan beberapa lapisan operasi XOR.