Encryption in Python

1 Introduction

In this project XOR cipher is used to encrypt and decrypt messages. The XOR cipher uses the bitwise XOR operation to encrypt and decrypt messages. The XOR cipher is symmetric key cryptography; it means the same operation is used to encode and decode the message i.e. the same key can be used to encode and decode the message.

1. Encryption: The encryption function *def encrypt(message,key)* takes a message to be encoded and the secret key as arguments. In the encryption process, XOR operation is performed on each pair of corresponding bits of message and the secret key.
2. Decryption: The decryption function *def decrypt(encoded_message,key)* takes an encoded message from the encryption function and the secret key as arguments. In the decryption process, XOR operation is performed on each pair of corresponding bits of encrypted message and key.

   Original message = Encrypted message $\oplus$ Key

   where $\oplus$ represents XOR operation

2 Code Snippets

Following are the code snippets for the encryption and decryption functions:

```
[1] def encrypt(message, key):
      key_bytes=(key*(len(message) // len(key) +1)).encode('utf-8')
      message_bytes = message.encode('utf-8')

      transformed_bytes = bytes(mb ^ kb for mb, kb in zip(message_bytes, key_bytes))
      encoded_message = transformed_bytes.decode('utf-8')
      return encoded_message

    encoded_message= encrypt('Apple','10001010101')
    print(encoded_message)
```

```
p@@\T
```

```
def decrypt(encoded_message,key):
  encoded_bytes = encoded_message.encode('utf-8')

  key_bytes = (key * (len(encoded_bytes) // len(key) + 1)).encode('utf-8')
  message_bytes = bytes(tb ^ kb for tb, kb in zip(encoded_bytes, key_bytes))

  original_message = message_bytes.decode('utf-8')
  return original_message
decrypt(encoded_message,'10001010101')
```

```
'Apple'
```

3 Analysis

The XOR cipher is a type of symmetric key cipher where the same key is used for encryption and decryption. So the security of the XOR cipher relies on the secrecy of the key.

The eavesdropping effect on the XOR cipher is significant. Suppose Eve is able to obtain the secret key used to encode the message sent by Alice to Bob. Now Eve can easily decrypt the cipher text and access the original message.

But if the key is shared through QKD, the communication becomes more secure. For instance the key generated by Alice can be shared using BB84 protocol. Alice creates qubits for her key and sends it through the quantum channel to Bob. Bob then creates his own key. Finally Alice and Bob compare keys and check for an eavesdropper.

4 Conclusion

The development of quantum cryptography is important for the future of secure communication due to following reasons:

- Quantum encryption implemented through QKD provides enhanced security.
- Quantum Encryption helps to detect any eavesdropper on the communication channel.
- Quantum computers have potential to break many existing widely used cryptography algorithms. So, the development of quantum encryption is a must for the protection against the quantum computing threats.