33_IT_A_MOHIT GUPTA

Cryptography & Network Security

TE IT

Experiment No 1 (Part B)

Implementation of Vernam Cipher (One-Time-Pad)

Aim: To write a Python program to perform encryption using conventional cryptography technique – Vernam Cipher.

Learning Objectives:

- To understand the transposition technique.
- To describe the Vernam cipher algorithm and encrypt the plain text.
- To decrypt the cipher text using Vernam cipher algorithm.

Theory:

Transposition Techniques:

Substitution techniques focus on substituting a plain-text alphabet with a cipher-text alphabet. Transposition techniques differ from substitution techniques in the way that they do not simply replace one alphabet with another, but they also perform some permutation over the plain text.

Algorithm of Vernam Cipher (One-Time Pad):

The Vernam cipher, whose specific subset is called one-time pad, is implemented using a random set of non-repeating characters as the input cipher text. The most significant point here is that once an input cipher text for transposition is used, it is never used again for any other message (hence the name one-time).

The length of the input cipher text is equal to the length of the original plain text. The algorithm used in the Vernam cipher is described in Fig. 1

1. Treat each plain-text alphabet as a number in an increasing sequence, i.e. A = 0, B = 1, … Z = 25.
2. Do the same for each character of the input cipher text.
3. *Add* each number corresponding to the plain-text alphabet to the corresponding input cipher-text alphabet number.
4. If the sum thus produced is greater than 26, subtract 26 from it.
5. Translate each number of the sum back to the corresponding alphabet. This gives the output cipher text.

Fig. 1 Algorithm for Vernam cipher

It should be clear that since the one-time pad is discarded after a single use, this technique is highly secure and suitable for small plain-text messages, but is clearly impractical for large messages.

Results:

Encrypt and Decrypt your full name using Vernam Cipher using. Assume a suitable one-time pad for encryption and decryption.

33_IT_A_MOHIT GUPTA

Conclusion:

In conclusion, the Vernam cipher, also known as the one-time pad, offers perfect secrecy when used correctly. It encrypts messages by combining a random key of the same length as the plaintext, resulting in statistically random ciphertext. However, the practical limitations of generating, managing, and securely distributing long keys make it suitable only for short, critical communications where absolute secrecy is essential. For general-purpose encryption of longer messages, other cryptographic algorithms are more practical and widely used.

Answer the following questions.

Why is Vernam Cipher suitable only for short messages?

The Vernam cipher requires a key as long as the plaintext, making it impractical for longer messages. Key generation and distribution complexities also limit its use for everyday communication. It's mainly suitable for short, critical communications when perfect secrecy is crucial.

33_IT_A_MOHIT GUPTA

Outputs:

verman.py > ...

```python
def vernam_encrypt(key, message):
    message = message.upper()
    key = key.upper()

    if len(message) != len(key):
        raise ValueError("Message and key must have the same length.")

    alpha = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    result = ""

    for m, k in zip(message, key):
        if m in alpha and k in alpha:
            # Find the corresponding ciphertext letter in the alphabet using XOR operation
            # Convert letters to numerical values (A=0, B=1, ..., Z=25)
            m_val = alpha.find(m)
            k_val = alpha.find(k)

            # Perform XOR operation and wrap around if needed
            encrypted_val = (m_val + k_val) % len(alpha)

            # Find the corresponding encrypted letter
            result += alpha[encrypted_val]
        else:
            result += m

    return result
```

```python
def main():
    word = input("Enter the message to be encrypted (alphabets only): ")
    key = input("Specify the key value (should be the same length as the message): ")

    # Check if the input contains only alphabets
    if not word.isalpha() or not key.isalpha():
        print("Please enter alphabets only.")
        return

    encrypted = vernam_encrypt(key, word)
    print("Encrypted message:", encrypted)

if __name__ == "__main__":
    main()
```

```
C:\Users\Rohit D-Roxx\Desktop\HTMLS\pythons>C:/Python311/python.exe "c:/Users/Rohit D-Roxx/Desktop/HTMLS/pythons/verman.py"
Enter the message to be encrypted (alphabets only): mohitgupta
Specify the key value (should be the same length as the message): RTMNYLZUYF
Encrypted message: DHTVRRTJRF

C:\Users\Rohit D-Roxx\Desktop\HTMLS\pythons>
```

33_IT_A_MOHIT GUPTA

Practical Learning Outcomes:

After performing the practical, the learner is able to:	Marked √
i. To understand the transposition technique. ii. To describe the Vernam cipher algorithm and encrypt the plain text. iii. To decrypt the cipher text using the Vernam cipher algorithm.	

Outcome	PLO 1	PLO 2	PLO 3	Performance	Attendance	Total Score	IT DEPARTMENT- TCET
Weight	**20**	**20**	**20**	**20**	**20**	**100**	Date of Performance: _____ Date of Correction: _____
Score							Roll No: _____ Marks: _____/100 Signature of Faculty: