Лабораторная работа № 8

Дисциплина: Информационная безопасность

Сулицкий Богдан Романович

Содержание

Список иллюстраций

1 Цель работы

Целью данной лабораторной работы является освоение на практике применения режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

2 Выполнение лабораторной работы

1. Я добавил нужные библиотеки для дальнейших действий(2.1).

```
import random
import string
```

Рис. 2.1: Добавление библиотек

2. Я создал класс VernamCipher, принимающий в конструкторе текст как переменную(2.2).

```
class VernamCipher:
    def __init__(self, t, key=None):
        self.P = t
        self.len = len(t)
        self.alf = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя" + string.ascii_lowercase + string.digits
        if key is None:
            self.K = self.key_create()
        else:
            self.K = key
        self.C = self.coder(self.P, self.K)
```

Рис. 2.2: Класс и конструктор

3. Я создал метод, генерирующий ключ шифрования(2.3).

```
def key_create(self):
    return "".join(random.choice(self.alf) for i in range(self.len))
```

Рис. 2.3: Метод генерации ключа шифрования

4. Я создал метод, который не только шифрует, но и дешифрует текст(2.4).

```
def coder(self, line, key):
    return "".join(chr(ord(c) ^ ord(k)) for c, k in zip(line, key))
```

Рис. 2.4: Метод шифровки/дешифровки текста

5. Я создал метод, который выводит исходный текст, ключ шифрований и шифротекст(2.5).

```
def print_all(text):
    print("Текст:", text.P, "\nКлюч:", text.K, "\nШифротекст:", text.C)
```

Рис. 2.5: Метод вывода

6. Я создал функцию нахождения исходного текста(2.6).

```
def find_text(cypher, texts, s):
    possible_keys = []
    for f in range(len(texts)):
        for i in range(len(cypher[f]) - s + 1):
            key = [chr(ord(c) ^ ord(k)) for c, k in zip(cypher[f][i:i + s], texts[f])]
            intact_plaintext = text1.coder(cypher[f], key)
            if texts[f] in intact_plaintext:
                possible_keys.append(''.join(key))
    rez = [''.join(chr(ord(c) ^ ord(k)) for c, k in zip(t, possible_keys[-1])) for t in cypher]
    return rez
```

Рис. 2.6: Функция нахождения

7. Я создал инициализацию класса с вводом текста и вызов всех методов класса с последующим выводом данных(2.7).

```python
if __name__ == "__main__":
    text1 = VernamCipher()  # Первое слово
    text2 = VernamCipher(text1.K)  # Второе слово
    print_all(text1)
    print_all(text2)
    D = find_text( cypher: [text1.C, text2.C],  texts: [text1.P, text2.P], text1.len)
    print("Декодированный текст 1:", D[0],
          "\nДекодированный текст 2:", D[1])
```

Рис. 2.7: Инициализация класса и вывод

8. Результат (2.8)

```
Введите открытый текст: Содержимое первого текста
Введите открытый текст: Содержимое второго текста
Текст: Содержимое первого текста
Ключ: аогксаб3вz6пóqbырс4skцд4кd
Шифротекст: ▫▫ёц▫▫▫▫ї  Ц▫▫яБ▫▫Ґбѐuуй▫▫бў|uў
D
Текст: Содержимое второго текста
Ключ: аогксаб3вz6пóqbырс4skцд4кd
vяTuуй▫▫бў|uў

Декодированный текст 1: Содержимое первого текста
Декодированный текст 2: Содержимое второго текста
```

Рис. 2.8: Результат работы программы

3 Вывод

В ходе проделанной лабораторной работы я освоил на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Список литературы

[1] https://esystem.rudn.ru/pluginfile.php/2090286/mod_resource/content/2/008-lab_crypto-key.pd