Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления

Кафедра Интеллектуальных информационных технологий

Отчет по лабораторной работе №2

Вариант №12

по дисциплине

Средства и методы защиты информации в интеллектуальных системах

Выполнил: А. М. Рутковский

Студент группы

121703

Проверил: В. В. Захаров

Минск 2023

Тема: Простейшие криптографические преобразования

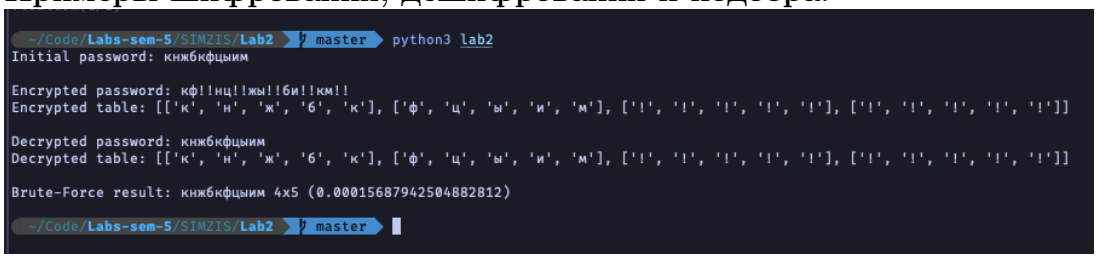
Задачи:

- Реализовать в виде программы шифр перестановки, использующий простые (прямоугольные) таблицы.
- Реализовать в виде программы атаку полным перебором ключа, используя для оценки правильности выбора ключа визуальный метод или исходный текст для автоматического сравнения результата дешифрования.
- Оценить криптографическую стойкость реализованного шифра.
- Предложить варианты усложнения шифра. Предложенные варианты оформить в виде алгоритма.

Листинг программы:

```python
import random
import time
import math


ALPHABET = "абвгдеёжзийклмнопрстуфхцчшщъыьэюя"

PASSWORD_LENGTH = 10

ROWS = 4
COLS = 5

def generate_random_text(alphabet, length):
    generated_string = "".join(random.choice(alphabet) for _ in range(length))
    return generated_string

def encrypt(text, rows, columns):
    text = text.replace(" ", "")
    encryptedText = ""

    table = []
    for i in range(rows):
        row = []
        for j in range(i * columns, i * columns + columns):
            if j < len(text):
                row.append(text[j])
            else:
                row.append("!")
        table.append(row)

    for i in range(columns):
        for j in range(rows):
            encryptedText += table[j][i]

    return encryptedText.strip(), table

def decrypt(encrypted_text, rows, columns):
    decrypted_text = ""
    table = []

    for i in range(rows):
        row = []
        for j in range(i, len(encrypted_text), rows):
            row.append(encrypted_text[j])
        table.append(row)

    for i in range(rows):
        for j in range(columns):
            decrypted_text += table[i][j]

    return decrypted_text.strip().replace("!", ""), table

def brute_force(encrypted_text, text):
    max_rows = math.ceil(len(encrypted_text) / COLS)
    max_columns = math.ceil(len(encrypted_text) / ROWS)

    best_decryption = ""
    best_key = ""

    for rows in range(1, max_rows + 1):
        for columns in range(1, max_columns + 1):
            decrypted_text, _ = decrypt(encrypted_text, rows, columns)

            if decrypted_text.lower() == text.lower():
                best_decryption = decrypted_text
                best_key = f"{rows}x{columns}"
                break

    return best_decryption, best_key

if __name__ == "__main__":
    text = generate_random_text(ALPHABET, PASSWORD_LENGTH)

    print("Initial password:", text)

    print()

    encrypted_text, encrypted_table = encrypt(text, ROWS, COLS)
    decrypted_text, decrypted_table = decrypt(encrypted_text, ROWS, COLS)

    print("Encrypted password:", encrypted_text)
    print("Encrypted table:", encrypted_table)

    print()

    print("Decrypted password:", decrypted_text)
    print("Decrypted table:", decrypted_table)

    print()

    start = time.time()
    best_decryption, best_key = brute_force(encrypted_text, text)
    end = time.time()

    print("Brute-Force result:", best_decryption, best_key, f"({end-start})")


```

Примеры шифрования, дешифрования и подбора:



Результаты подбора

- 5 символов
  - Текст - **овфс**;
  - Шифртекст - **офвс**;
  - Результат подбора - 2x2 (<1мс);
- 8 символов
  - Текст - **пхзфяхдщ**;
  - Шифртекст - **пяххздфщ**;
  - Результат подбора - 2x4 (1мс);
- 20 символов
  - Текст - **чжхйжщргпйгюусцхжтвц**;
  - Шифртекст - **чщгхжрюжхгутйпсвжйцц**;
  - Результат подбора - 4x5 (2.3мс);

Идея усложнения алгоритма

Когда таблица построена, мы имеем возможность начать шифрование пароля не с первого столбца, а с определенного столбца под номером "n". Для этого требуется использовать ключ, который определит номер столбца, с которого будет идти начало шифрования. Пример: Возьмем слово **"Беларусь"**.

При стандартном шифровании используя таблицу **2x4** мы получим шифр: **Бреулсаь**

А если возьмем в качестве начального столбца не 0, а 1, то получим - **еулсаьБр**

Алгоритм

- Открытый текст построчно, начиная с верхней строки, вписать в таблицу состоящую из m строк и n столбцов.
- Задать ключ, который будет указывать номер столца, с которого нужно начинать щифрование. Номер столбца должен быть равен меньше или равен общему числе столбцов в таблице.
- Запишите символы из таблицы, начиная с выбранного столбца согласно ключу.

Вывод

При использовании атаки полного перебора ключа было обнаружено, что шифр перестановки с использованием простых таблиц имеет низкую стойкость криптографии. Это объясняется тем, что использование простых таблиц ограничивает количество возможных ключей, что делает относительно легким восстановление исходного текста. Однако, с увеличением размеров таблицы время, необходимое для дешифрования, также увеличивается. Для обеспечения более высокой стойкости шифра необходимо внести дополнительные модификации