Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет информационных технологий и управления
Кафедра интеллектуальных информационных технологий
Дисциплина «Средства и методы защиты информации в интеллектуальных системах»

ОТЧЁТ
к лабораторной работе №2
на тему
«ПРОСТЕЙШИЕ КРИПТОГРАФИЧЕСКИЕ ПРЕОБРАЗОВАНИЯ»

БГУИР 6-05-0611-03 130

Выполнил студент группы 321701
СЕМЕНЯКО Владимир Дмитриевич

_____
(дата, подпись студента)

Проверил
САЛЬНИКОВ Даниил Андреевич

_____
(дата, подпись преподавателя)

Минск 2025

1 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

а) Разработать программу на языке Python, которая реализует шифр Виженера, включая функции зашифрования и расшифрования для английского текста.

б) Реализовать программу для проведения атаки на шифр Виженера методом полного перебора ключа. Оценка правильности ключа должна производиться путём поиска общеупотребительных слов в расшифрованном тексте.

в) Провести оценку криптографической стойкости шифра Виженера, указав его основные уязвимости.

г) Предложить и описать в виде алгоритмов варианты усложнения данного шифра для повышения его надёжности.

2 ВЫПОЛНЕНИЕ РАБОТЫ

Программа была реализована на языке Python и состоит из двух основных функций: vigenerecipher() для шифрования/расшифрования и bruteforceattack() для взлома шифротекста. Программа демонстрирует как процесс шифрования, так и атаку на зашифрованное сообщение.

Листинг 1 – Код программы

```
import itertools


def vigenere_cipher(text: str, key: str, decrypt: bool = False) -> str:
    key = key.lower()
    if not key.isalpha():
        raise ValueErrorКлюч(” должен состоять только из букв!”)

    result_chars = []
    key_index = 0
    for char in text:
        if ’a’ <= char.lower() <= ’z’:
            shift = ord(key[key_index % len(key)]) - ord(’a’)
            if decrypt:
                shift = -shift

            start_char_code = ord(’A’) if char.isupper() else ord(’a’)
            processed_char_code = (ord(char) - start_char_code + shift) % 26
            result_chars.append(chr(start_char_code + processed_char_code))
            key_index += 1
        else:
            result_chars.append(char)
    return ””.join(result_chars)
```

```
def brute_force_attack(ciphertext: str, max_key_length: int) -> dict:
    common_words = [”the”, ”and”, ”for”, ”are”, ”but”, ”not”, ”you”, ”all”, ”
   was”, ”her”]
    alphabet = ”abcdefghijklmnopqrstuvwxyz”

    for length in range(1, max_key_length + 1):
        for key_tuple in itertools.product(alphabet, repeat=length):
            key = ””.join(key_tuple)
            decrypted_text = vigenere_cipher(ciphertext, key, decrypt=True)
            if any(f” {word} ” in decrypted_text.lower() for word in
   common_words):
                return {”key”: key, ”decrypted_text”: decrypted_text}
    return {”key”: None, ”decrypted_text”: Не” удалось найти ключ”}
```

Программа успешно выполняет шифрование и расшифрование текста. Функция атаки методом полного перебора эффективно находит короткие ключи (обычно длиной до 4-5 символов) за приемлемое время, сканируя результаты расшифрования на наличие известных английских слов.

```
Оригинальный текст:  The Vigenere cipher is a polyalphabetic substitution cipher.
Ключ:            secret
Зашифрованный тектс: Llg Mmzwrgii vatjvv bk e rfprspryeuwxkt wntwvzxnlmqe gbhlgi.
Расшифрованный текст: The Vigenere cipher is a polyalphabetic substitution cipher.

--- Brute-Force Атака ---
Текст подверженный атаке: Dlgc mq k wcmvcd qccwyqi rrer gi uspj dvw ds zbiyu.
Успех! Найден ключ: 'cbd'
Расшифрованное сообщение:  'Bkda ln i vzkuzb pzavvoh opdo eh rqog but br wzhvs.'
```

Рисунок 1 – Результат выполнения программы

Шифр Виженера является значительным шагом вперёд по сравнению с моноалфавитными шифрами, так как он противостоит простому частотному анализу. Однако его криптографическая стойкость полностью зависит от длины и случайности ключа.

Ключевое пространство шифра определяется формулой:

$$N = M^L$$

Рисунок 2 – Формула для определения ключевого пространства шифра

Несмотря на это, шифр имеет критическую уязвимость - повторяющийся ключ. Если ключ короткий, он будет повторяться. Это создаёт периодичность в шифротексте, которую можно обнаружить. Для повышения криптографической стойкости шифра Виженера необходимо устранить его главную слабость.

а) **Использование бегущего ключа (Running Key)**

Вместо короткого, повторяющегося ключа используется длинный ключ, сопоставимый по длине с самим сообщением (например, текст из книги). Это исключает цикличность и делает метод Касиски неприменимым.

Алгоритм:

– **Ключ:** Выбирается текст K (например, из книги), длина которого не меньше длины исходного сообщения P.

– **Зашифрование:** Символ шифротекста C_i получается путём сдвига символа P_i на величину, соответствующую символу ключа K_i.

– **Результат:** Полностью устраняется периодичность, что делает метод Касиски неприменимым.

б) **Шифр с автоключом (Autokey Cipher)**

Ключ генерируется на основе самого исходного сообщения, что усложняет криптоанализ.

Алгоритм:

– **Инициализация:** Выбирается короткий начальный ключ K_{seed}.

– **Генерация полного ключа:** Полный ключ K формируется как конкатенация начального ключа и самого исходного текста: $K = K_{seed} + P$.

– **Зашифрование:** Шифрование выполняется стандартным методом Виженера, но с использованием сгенерированного длинного ключа K.

в) **Добавление перестановок**

Комбинация шифра Виженера с перестановочным шифром может значительно ”перемешать”статистические закономерности.

Алгоритм:

– **Шифрование Виженера:** Исходный текст P шифруется с помощью ключа K_1 для получения промежуточного шифротекста C_1.

– **Перестановка:** Символы в C_1 переставляются согласно правилу, заданному вторым ключом K_2 (например, по столбцам в таблице).

– **Результат:** Частотные характеристики шифротекста сильно искажаются, что затрудняет криптоанализ.

ВЫВОД

В ходе лабораторной работы был успешно реализован шифр Виженера и атака на него. Анализ показал, что, несмотря на историческую значимость, в своей классической форме шифр уязвим при использовании коротких ключей. Предложенные методы усложнения, такие как использование ”бегущего”ключа или автоключа, способны значительно повысить его криптографическую стойкость за счёт устранения периодичности ключа.