Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина: Методы защиты информации

ОТЧЕТ
по лабораторной работе №3
на тему

Асимметричная криптография. Криптосистема Рабина

Студент Д. А. Демидова

Проверил Е. А. Лещенко

Минск 2024

СОДЕРЖАНИЕ

1 ЦЕЛЬ РАБОТЫ

В данной лабораторной работе необходимо изучить теоретические сведения, а также реализовать программное средство шифрования и дешифрования текстовых файлов при помощи Криптосистемы Рабина.

2 РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ

Содержимое файла input.txt:
hello my name is dasha
i'm studying in bsuir
lalalalalalalalalalala

Содержимое файла encrypted.txt:
3774739762831
2642585648526
6033595269364
6033595269364
2149036078850
11302217046933
10825907921335
11101540786463
11302217046933
1343470409819
5976155740918
10825907921335
2642585648526
11302217046933
4259169708665
10000317184308
11302217046933
6466038408829
5976155740918
10000317184308
3774739762831
5976155740918
10949279341278
4259169708665
7263865958438
10825907921335
11302217046933
10000317184308
839015395989
1965716788217
6466038408829
11101540786463
4259169708665
1343470409819
1296936325527

11302217046933
4259169708665
1343470409819
11302217046933
8132823455025
10000317184308
1965716788217
4259169708665
4886868809140
10949279341278
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918
6033595269364
5976155740918

Содержимое файла decrypted.txt:
hello my name is dasha
i'm studying in bsuir
lalalalalalalalalalala

ВЫВОД

В ходе выполнения данной лабораторной работы были изучены теоретические сведения об алгоритме шифрования при помощи Криптосистемы Рабина.

Также было реализовано консольное приложение, осуществляющее шифрование и дешифрование содержимого текстовых файлов ппри помощи Криптосистемы Рабина.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг программного кода

Файл main.py

```
import rabin as rb


def read_file(filename: str):
    with open(filename, 'r') as file:
        result = file.read()
        file.close()
    return result


def read_file_lines(filename: str):
    with open(filename, 'r') as file:
        result = file.readlines()
        file.close()
    return result


def write_file(filename: str, value):
    with open(filename, 'w') as file:
        if type(value) is list:
            if type(value[0]) is list:
                for v in value:
                    file.writelines(v)
        else:
            file.writelines(value)

        file.close()


def get_max(text: list) -> int:
    max_len = 0

    for temp in text:
        for t in temp:
            value = t.bit_length()
            if max_len < value:
                max_len = value

    return max_len


def rabin_processing(input: str):
    input_text = read_file(f'{input}.txt')
```

```
    text = [
        [
            text[i:i + 32] for i in range(0, len(text), 32)
        ]
        for text in input_text]

    i_text = [
        [rb.text_to_int(t_iter) for t_iter in t]
        for t in text]

    max_bit_len = get_max(i_text)
    p, q = rb.get_keys(max_bit_len)
    n = p * q

    enc_int = [
        [
            f'{rb.encrypt_text(it, n)}\n' for it in temp
        ]
        for temp in i_text]

    output = f'encrypted-{input}.txt'
    write_file(output, enc_int)

    enc_int = read_file_lines(output)
    enc_int = [int(elem) for elem in enc_int]

    solutions = [rb.decrypt_text(enc, p, q, n) for enc in enc_int]
    results = [rb.find_correct_solution(sol) for sol in solutions]

    output_res = []
    for res in results:
        if res is not None:
            output_res.append(
                rb.int_to_text(res >> 8))

    write_file(f'decrypted-{output}', ''.join(output_res))


def main():
    rabin_processing('text')


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

Файл rabin.py

```
import random
```

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True


def generate_prime_number(bits):
    while True:
        num = random.getrandbits(bits)
        if num % 4 == 3 and is_prime(num):
            return num


def get_keys(bits: int) -> tuple:
    p = generate_prime_number(bits)
    q = generate_prime_number(bits)

    while p == q:
        q = generate_prime_number(bits)

    return p, q


def text_to_int(text: str) -> int:
    res = 0

    for i in range(len(text)):
        res = (res + ord(text[i])) << 16

    return res


def int_to_text(value: int) -> str:
    res = []
    while value != 0:
        res.append(chr((value) & 0xFFFF))
        value >>= 16

    res = res[1:]
    return ''.join(res[::-1])


def encrypt_text(p, n):
    p <<= 8
    p |= 0xFF
    return pow(p, 2, n)


def extended_euclidean(p, q):
```

```
    if p == 0:
        return q, 0, 1
    else:
        gcd, y, x = extended_euclidean(q % p, p)
        return gcd, x - (q // p) * y, y


def decrypt_text(c, p, q, n):
    ext_eucl = extended_euclidean(p, q)

    a, b = ext_eucl[1], ext_eucl[2]

    r = pow(c, (p + 1) // 4, p)
    s = pow(c, (q + 1) // 4, q)

    x = int((a * p * s + b * q * r) % n)
    x_1 = n - x
    y = int((a * p * s - b * q * r) % n)
    y_1 = n - y

    return x, x_1, y, y_1


def find_correct_solution(solutions: list) -> int:
    bit_mask = (1 << 8) - 1
    for sol in solutions:
        n = sol & bit_mask
        if n == 0xFF:
            return sol

    return None
```