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 3](#__RefHeading___Toc7947_2113241753)

[2 Результат работы программы 4](#__RefHeading___Toc7949_2113241753)

[Вывод 5](#__RefHeading___Toc7951_2113241753)

[Приложение A](#__RefHeading___Toc7959_2113241753) [(обязательное)](#__RefHeading___Toc7961_2113241753) [Листинг программного кода 6](#__RefHeading___Toc7963_2113241753)

**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 A**

**(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