МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
Новоуральский технологический институт –
филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»
(НТИ НИЯУ МИФИ)
Колледж НТИ

Цикловая методическая комиссия информационных технологий

ОТЧЕТ №10

ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ НА ТЕМУ

«ПРОГРАММНАЯ РЕАЛИЗАЦИЯ КРИПТОГРАФИЧЕСКИХ АЛГОРИТМОВ»

ПМ.05 «Разработка программного обеспечения компьютерных сетей»
МДК.05.01 «Защита информации в КС»

Специальность СПО 09.02.03
«Программирование в компьютерных системах»

очная форма обучения
на базе основного общего образования

Выполнил
студент группы КПР–47 Д
Егорушкин И.А. 11.12.2020 
дата подпись

Проверил
преподаватель _____ _____
Горницкая И.И. дата подпись

Новоуральск 2020

Цель работы: Анализ рисков информационной безопасности

Оборудование:

AMD Ryzen 5 3550U

ОЗУ 8 Гб

Программное обеспечение:

Windows 10 Professional 64 бит;

Ход работы:

Номер варианта	**Исходные данные**							
	Часть 1	**Часть 2**						
	Алгоритм шифрования	**p**	**q**	**e**	**d**	**m_1**	**m_2**	**m_3**
2	Одиночная перестановка	17	11	7	23	8	15	45

Применение алгоритма симметричного шифрования

Симметричное шифрование используется для обмена данными во многих современных сервисах, часто в сочетании с асимметричным шифрованием. Например, мессенджеры защищают с помощью таких шифров переписку (при этом ключ для симметричного шифрования обычно доставляется в асимметрично зашифрованном виде), а сервисы для видеосвязи — потоки аудио и видео. В защищенном транспортном протоколе TLS симметричное шифрование используется для обеспечения конфиденциальности передаваемых данных.

Симметричные алгоритмы не могут применяться для формирования цифровых подписей и сертификатов, потому что секретный ключ при использовании этого метода должен быть известен всем, кто работает с шифром, что противоречит самой идее электронной подписи (возможности проверки ее подлинности без обращения к владельцу).

Применение алгоритма асимметричного шифрования

Асимметричное шифрование решает главную проблему симметричного метода, при котором для кодирования и восстановления данных используется один и тот же ключ. Если передавать этот ключ по незащищенным каналам, его могут перехватить и получить доступ к зашифрованным данным. С другой стороны, асимметричные алгоритмы гораздо медленнее симметричных, поэтому во многих криптосистемах применяются и те и другие.

Например, стандарты SSL и TLS используют асимметричный алгоритм на стадии установки соединения (рукопожатия): с его помощью кодируют и передают ключ от симметричного шифра, которым и пользуются в ходе дальнейшей передачи данных.

Также асимметричные алгоритмы применяются для создания электронных подписей для подтверждения авторства и (или) целостности данных. При этом подпись генерируется **с** помощью закрытого ключа, а проверяется с помощью открытого.

Одиночная перестановка

```
def encode(keyword, message, normalize=False):
    # True - отбрасывать пробелы при шифровании
    if normalize:
        message = ''.join(message.split())

    rows = len(message) // len(keyword)
    if len(message) % len(keyword) != 0:
        rows += 1

    indexes = sorted([(index, value) for index, value in enumerate(keyword)], key=lambda item:
item[1])
    result = ''

    for row in range(rows):
        for index in indexes:
            position = index[0] * rows + row
            if position < len(message):
                result += message[position]
            else:
                result += ' '

    return result



def decode(keyword, cipher):
    rows = len(cipher) // len(keyword)
    if len(cipher) % len(keyword) != 0:
```

```
        rows += 1

    indexes = sorted([(index, value) for index, value in enumerate(keyword)], key=lambda item: item[1])
    indexes = sorted([(index, value) for index, value in enumerate(indexes)], key=lambda item: item[1][0])
    result = ''

    for index in indexes:
        for row in range(rows):
            position = index[0] + len(keyword) * row
            if position < len(cipher):
                result += cipher[position]

    return result


key = 'Илья'
text = 'Егшорушкин Илья Андреевич'

enc = encode(key, text)
print('ENCODE:', enc)
dec = decode(key, enc)
print('DECODE:', dec)
```

Программа шифрования и дешифрования сообщения при помощи алгоритма RSA

```
p = 17
q = 11
e = 7
d = 23
m1 = 8
m2 = 15
m3 = 45


def rsa(p, q, e, d, m):
```

```
    print('сообщение', m)
    n = p * q
    Fq = (p - 1) * (q - 1)
    c = m ** e % n
    print('ENCODE', c)

    c = c ** d % n
    print('DECODE:', c)
    print()


rsa(p, q, e, d, m1)
rsa(p, q, e, d, m2)
rsa(p, q, e, d, m3)
```

Результаты шифрования и дешифрования заданных сообщений

сообщение 8
ENCODE 134
DECODE: 8

сообщение 15
ENCODE 93
DECODE: 15

сообщение 45
ENCODE 122
DECODE: 45

Вывод : были применены знания шифрации и дешифрации сообщений в виде кода.