Правительство Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
(НИУ ВШЭ)

Московский институт электроники и математики им. А.Н. Тихонова

ОТЧЕТ
О ПРАКТИЧЕСКОЙ РАБОТЕ № 2
по дисциплине «Криптографические методы защиты информации»
ТЕМА РАБОТЫ

Современные симметричные шифры

Студент гр. ____
Е.В. Шараев
«24» марта 2024 г.

Руководитель
Заведующий кафедрой информационной безопасности киберфизических систем
канд. техн. наук, доцент
_____О.О. Евсютин
«___» _____ 2024 г.

Москва 2024

СОДЕРЖАНИЕ

1 Задание на практическую работу

Целью данной работы является приобретение навыков программной реализации современных алгоритмов симметричного шифрования.

В рамках практической работы необходимо выполнить следующее:

1 написать программную реализацию одного из следующих симметричных шифров (по выбору студента):

- Магма;
- **Кузнечик; (выбранный вариант)**
- AES;

2 подготовить отчет о выполнении работы.

Программа должна обладать следующей функциональностью:

- принимать на вход файл, содержащий открытый текст, подлежащий зашифрованию, или шифртекст, подлежащий расшифрованию;
- принимать на вход секретный ключ;
- [***дополнительная опция, не являющаяся обязательной***] давать пользователю возможность выбирать режим работы блочного шифра;
- осуществлять зашифрование или расшифрование выбранного файла по выбору пользователя и сохранять результат в новом файле.

Отчет должен содержать следующие составные части:

- ✓ раздел с заданием;
- ✓ раздел с краткой теоретической частью;
- ✓ раздел с описанием программной реализации с учетом особенностей выбранной среды разработки и языка программирования;
- ✓ раздел с результатами работы программы;
- ✓ раздел с выводами о проделанной работе.

2 Краткая теоретическая часть

«Кузнечик» (англ. Kuznyechik[1] или англ. Kuznechik[2][3]) — симметричный алгоритм блочного шифрования с размером блока 128 бит и длиной ключа 256 бит, использующий для генерации раундовых ключей SP-сеть.

Описание алгоритма. Для шифрования, расшифрования и генерации ключа используются следующие функции:

1. XOR с раундовым ключом
2. Нелинейное биективное преобразование **S** (подстановка по таблице соотвестствия)
3. Линейное преобразование **L**, где происходит сдвиг элементов блока на 1 блок, а утраченный после сдвига блок восполняется путем свертки всех блоков в один в результате линейного преобразования

При зашифровке операции XSL производятся 9 раз (раундов), а 10-й раунд включает только операцию наложения раундового ключа. Расшифрование представляет собой последовательное применение обратных процедур. Подборнее алогитм описывается в Части 3

Режим работы.

Режимы работы гаммирования (реализованный мною помимо простой подстановки блоков) подразумевает не прямое шифрование блоков открытого текста , а сгенерированного из синхропосылки счетчика (получение *гаммы*), затем на каждом блоке открытого текста применяется операция XOR с поученной *гаммой*, результат чего и попадает в ширф-текст.

3. Описание программной реализации

3.1 Описание блочного шифра Кузнечик

Программа реализована в классе *Cricket* в файле *cricket.py* в данном репозитории.

* Статические параметры *pi* и pi_*inv*: Целочисленные массивы, где индекс каждого элемента соответствует значению исходного при биективном нелинейном отображении (X) и обратной опреации (X^{-1}).

* Метод *__init__(self)* Запускается при инициализации класса. Принимает на вход главный ключ (256 bit), запускает генерацию раундовых ключей и сохраняет их в виде упорядоченного массива в параметре объекта *self.round_keys*

* Статический метод *__generate_round_keys(key)* Принимает на вход главный 256 битный ключ и генерирует из него 10 раундовых 128 битных ключей.

а) На первом метод разбивает глвный ключ на две равные части, и сохраняет их в массив *self.round_keys* в качестве первых двух ключей.

б) Остальные 8 (4 пары) ключей вырабатываются в цикле for i in range(4):, где на каждом шаге цикла к предыдущей паре ключей 8-кратно применяются преобразования сети Фейстеля.

* Метод *encrypt(self, x)*: Принимая на вход блок длинной 128 бит, затем в цикле for rnd in range(9): выполняет 9 раундов зашифрования.

а) Запускает сначала операцию **X** (XOR) с райндовым ключом

x ^ self.round_keys[rnd]

б) Затем нелинейное биективное преобразование **S**

Cricket.__s_transformation(x ^ self.round_keys[rnd])

в) И наконец линейное преобразование **L**

Cricket.__l_transformation(Cricket.__s_transformation(x ^ self.round_keys[rnd])) где каждый байт 16 раз помножается на соответствующий ему элемент поля Галуа. В программе этот шаг реализован в методе *__linear_function(x)*

г) Последний, 10 раунд зишифрования является не полным и состоит только из наложения последнего раундового ключа

return x ^ self.round_keys[-1]

* Расшифрование *def decrypt(self, x)* устроено противоположным образом. Метод принимает на вход зашифрованный блок длинной 128 бит.

а) Сначала строится развернутый массив ключей

keys = self.round_keys[::-1]

б) Затем 9 раз применяются обратные функции X^{-1} $S^{-1}L^{-1}$

x = Cricket.__s_inv_transformation(Cricket.__l_inv_transformation(x ^ keys[rnd]))

в) Последний раунд также является не полным и заключается лишь в применении последнего раундового ключа *return x ^ keys[-1]*

3.2 Описание режимов работы

Режимы реализованы в классе *EncryptionMode*. На момент составления отчета успел реализовать только метод простой подстановки (в методе *ecb_mode*) и метод гаммирования (*ctr_mode*)

* В режиме *ecb_mode* блоки открытого текста напрямую шифруются шифром Кузнечик.

Размер блока всегда 16 байт (128 бит) *block_size = 16*

Статический метод *__padding_bytes* сначал добавляет 1 (b'\x01'), а затем нулями добивает количество байт до кратного 16-ти

Шифрования блоков происходит последовательно и независимо друг от друга, поэтому в шифртекст переносятся статистические характеристики исходного текста, а значит этот шифр является не надежным

* В режиме Гаммирования шифрование происходит не напрямую. Вместо открытого текста алгоритм блочного шифра шифрует счетчик, состоящий из синхропосылки и нулей (на первом этапе), а затем усекается на заданное количество байт. В моей реализации я по умолчанию задаю размер блока 13 байт *block_size: int = 13*, но оно может быть изменено при вызове функции.

Гамма считается как результат шифрования счетчика, а затьем усечается на заданное количество байт *gamma = cricket.encrypt(counter) >> right_shift*

Потом я увеличиваю счетчик

counter = EncryptionMode.__increment_counter(counter)

И накладываю полученную гамму на блоки открытого текста *encrypted_block = gamma ^ block_int*

В конце конкатенирую полученные данные к коллекции *result_bytes.extend(encrypted_block)*

Метод *__get_counter* генерирует счетчик (размером 128 бит) из синхнопосылки и нулей

Метод *__increment_counter* очевидно служит для увеличения значения счетчика по мере шифрования.

Расшифрование происходит точно также, за исключением: того что на первом этапе нужно отделить синхропосылку и переопределить счетчик, а также удалить нулевые байты в конце до единичного байта (включительно)

4. Демонстрация работы программы

1) Чтобы запустить код выполните следующие действия:

./cricket.py <command> <mode> <path/to/file> <key>

, где:

<command>: --encrypt - чтобы зашифровать файл, --decrypt - чтобы расшифровать

<mode> - режим шифрования / расшифрования (--dummy / --counter)

<path/to/file> - путь к файлу. При зашифровании имя зашифрованного файла будет содержать дополнительное расширение .enc. При расшифровании - данное расширение, если оно имеется, будет удалено

<key> - 256-битный ключ в виде строки

По шагам:

Вывожу в консоль все файлы текущей директории

```
In [1]: # Выведем на консоль все файлы текущей директории
        !ls -l

        total 380
        -rwxrwxr-x 1 evgeny evgeny  15667 Mar 24 21:21 cricket.py
        -rw-rw-r-- 1 evgeny evgeny      0 Mar 24 20:50 Readme.md
        -rw-rw-r-- 1 evgeny evgeny  49457 Mar 24 21:59 report.ipynb
        -rw-rw-r-- 1 evgeny evgeny 311310 Mar 24 22:00 sharaev_evgeny_report_pr_2.pdf
        -rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 20:36 test.txt
```

Запускаю шифрование файла test.txt в режиме простой замены байт и смотрю что получилось

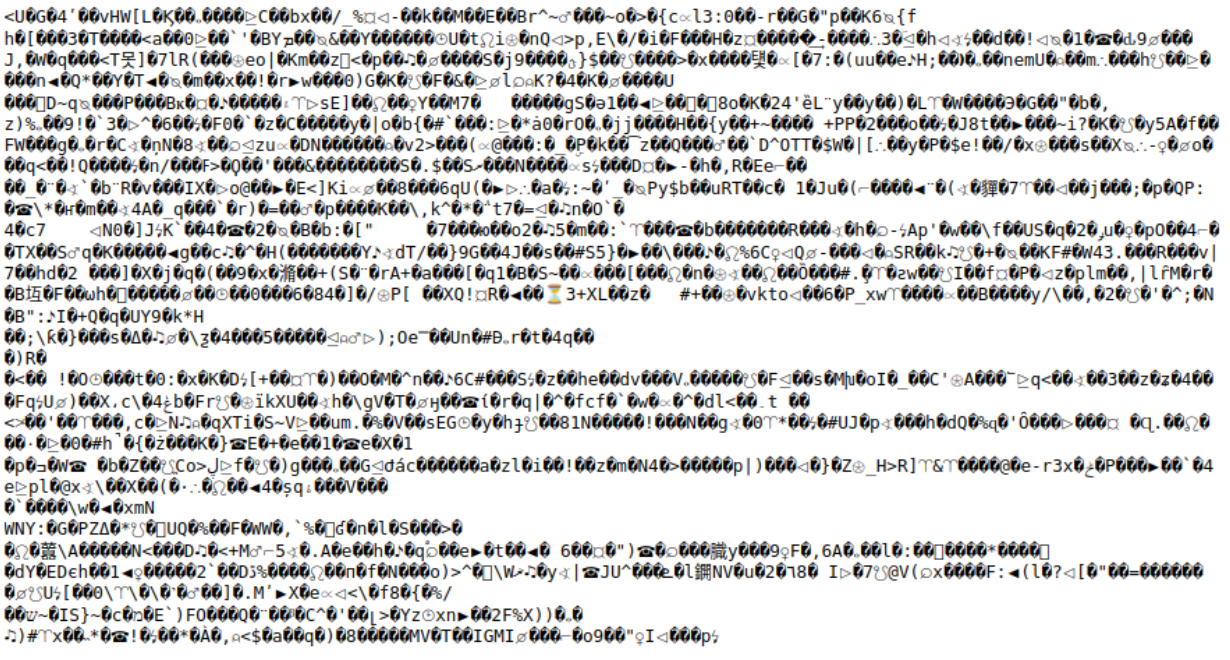
```
In [2]: # Запускаем зашифрование файла test.txt
        !./cricket.py --encrypt --dummy "test.txt" "8899aabbccddeeff0011223344556677fedcba98765432100123456789abcdef"

In [3]: !ls -l

        total 384
        -rwxrwxr-x 1 evgeny evgeny  15667 Mar 24 21:21 cricket.py
        -rw-rw-r-- 1 evgeny evgeny      0 Mar 24 20:50 Readme.md
        -rw-rw-r-- 1 evgeny evgeny  49457 Mar 24 21:59 report.ipynb
        -rw-rw-r-- 1 evgeny evgeny 311310 Mar 24 22:00 sharaev_evgeny_report_pr_2.pdf
        -rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 20:36 test.txt
        -rw-rw-r-- 1 evgeny evgeny   3312 Mar 24 22:50 test.txt.cricket
```

Вижу что появился файл test.txt.cricket. Посмотрим его содержимое

In [4]:
```
# Зашифрованный файл
!cat test.txt.cricket
```



А вот как выглядел исходный файл:

In [5]:
```
# Исходный файл
!cat test.txt
```

```
I
«Мой дядя самых честных правил,
Когда не в шутку занемог,
Он уважать себя заставил
И лучше выдумать не мог.
Его пример другим наука;
Но, боже мой, какая скука
С больным сидеть и день и ночь,
Не отходя ни шагу прочь!
Какое низкое коварство
Полуживого забавлять,
Ему подушки поправлять,
Печально подносить лекарство,
Вздыхать и думать про себя:
Когда же черт возьмет тебя!»
II
Так думал молодой повеса,
Летя в пыли на почтовых,
Всевышней волею Зевеса
Наследник всех своих родных.
Друзья Людмилы и Руслана!
С героем моего романа
Без предисловий, сей же час
Позвольте познакомить вас:
Онегин, добрый мой приятель,
Родился на брегах Невы,
Где, может быть, родились вы
Или блистали, мой читатель;
Там некогда гулял и я:
Но вреден север для меня 1.
III
Служив отлично благородно,
Долгами жил его отец,
Давал три бала ежегодно
И промотался наконец.
Судьба Евгения хранила:
```

Запускаю расшифровку файла и смотрю что получилось

```
In [6]: # Запускаю расшифровку файла
        !./cricket.py --decrypt --dummy "test.txt.cricket" "8899aabbccddeeff0011223344556677fedcba98765432100123456789abcdef
```

```
In [7]: # Смотрим
        !ls -l
```

```
total 388
-rwxrwxr-x 1 evgeny evgeny  15667 Mar 24 21:21 cricket.py
-rw-rw-r-- 1 evgeny evgeny      0 Mar 24 20:50 Readme.md
-rw-rw-r-- 1 evgeny evgeny  49457 Mar 24 21:59 report.ipynb
-rw-rw-r-- 1 evgeny evgeny 311310 Mar 24 22:00 sharaev_evgeny_report_pr_2.pdf
-rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 20:36 test.txt
-rw-rw-r-- 1 evgeny evgeny   3312 Mar 24 22:50 test.txt.cricket
-rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 22:50 test.txt.decrypted
```

Вижу что появился файл с расширением .decrypted. Посмотрим на его сожержимое

```
In [8]: !cat test.txt.decrypted
```

```
I
«Мой дядя самых честных правил,
Когда не в шутку занемог,
Он уважать себя заставил
И лучше выдумать не мог.
Его пример другим наука;
Но, боже мой, какая скука
С больным сидеть и день и ночь,
Не отходя ни шагу прочь!
Какое низкое коварство
Полуживого забавлять,
Ему подушки поправлять,
Печально подносить лекарство,
Вздыхать и думать про себя:
Когда же черт возьмет тебя!»
II
Так думал молодой повеса,
Летя в пыли на почтовых,
Всевышней волею Зевеса
Наследник всех своих родных.
Друзья Людмилы и Руслана!
С героем моего романа
Без предисловий, сей же час
Позвольте познакомить вас:
Онегин, добрый мой приятель,
Родился на брегах Невы,
Где, может быть, родились вы
Или блистали, мой читатель;
Там некогда гулял и я:
Но вреден север для меня 1.
III
Служив отлично благородно,
Долгами жил его отец,
```

Удалю все артифакты и запущу в режиме гаммирования

```
In [9]: # Удалю артефакты и вызову с новыми аргументами
        !rm -rf test.txt.encrypted test.txt.decrypted
```

```
In [10]: !./cricket.py --encrypt --control "test.txt" "8899aabbccddeeff0011223344556677fedcba98765432100123456789abcdef"
```

```
In [11]: !./cricket.py --decrypt --control "test.txt.cricket" "8899aabbccddeeff0011223344556677fedcba98765432100123456789abcd
```

```
In [12]: !ls -l
```

```
total 388
-rwxrwxr-x 1 evgeny evgeny  15667 Mar 24 21:21 cricket.py
-rw-rw-r-- 1 evgeny evgeny      0 Mar 24 20:50 Readme.md
-rw-rw-r-- 1 evgeny evgeny  49457 Mar 24 21:59 report.ipynb
-rw-rw-r-- 1 evgeny evgeny 311310 Mar 24 22:00 sharaev_evgeny_report_pr_2.pdf
-rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 20:36 test.txt
-rw-rw-r-- 1 evgeny evgeny   3323 Mar 24 22:50 test.txt.cricket
-rw-rw-r-- 1 evgeny evgeny   3302 Mar 24 22:50 test.txt.decrypted
```

Вновь образовалось два новых файла test.txt.cricket и test.txt.decrypted. Вот что в них содержится:

In [13]:
```
!cat test.txt.cricket
```

```
<illegible>
```

In [14]:
```
!cat test.txt.decrypted
```

```
I
«Мой дядя самых честных правил,
Когда не в шутку занемог,
Он уважать себя заставил
И лучше выдумать не мог.
Его пример другим наука;
Но, боже мой, какая скука
С больным сидеть и день и ночь,
Не отходя ни шагу прочь!
Какое низкое коварство
Полуживого забавлять,
Ему подушки поправлять,
Печально подносить лекарство,
Вздыхать и думать про себя:
Когда же черт возьмет тебя!»
II
Так думал молодой повеса,
Летя в пыли на почтовых,
Всевышней волею Зевеса
Наследник всех своих родных.
Друзья Людмилы и Руслана!
С героем моего романа
Без предисловий, сей же час
Позвольте познакомить вас:
Онегин, добрый мой приятель,
Родился на брегах Невы,
Где, может быть, родились вы
Или блистали, мой читатель;
Там некогда гулял и я:
```

5 Выводы о проделанной работе

Краткие выводы о проделанной работе.

6 Список использованных источников

ГОСТ 34.12 — 2015 Информационные технологии. Криптографическая защита информации. Блочный шифры

ГОСТ 34.13 - 2015 Информационные технологии. Криптографическая защита информации. Режимы работы блочных шифров