МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Мегафакультет компьютерных технологий и управления

Факультет безопасности информационных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ 1

«Алгоритмы криптографии и подпись приложений»

по дисциплине:

«Разработка систем аутентификации и криптографии»

Выполнила:

студент гр. N42514c,

Саенко Александра Игоревна

Проверил:

ассистент ФБИТ

Федоров Иван Романович

Санкт-Петербург

2020

Вариант: Blowfish

Цель работы: реализовать алгоритм шифрования Blowfish: процедуру генерации ключей, шифрования и дешифрования без использования криптографических библиотек. Программа должна запускаться в среде Windows, исполняемый файл программы должен иметь расширение .EXE. Подпись полученного файла .EXE с помощью команд Windows PowerShell.

Ход работы:

Blowfish представляет собой 64-битовый блочный шифр с ключом переменной длины. Алгоритм состоит из двух частей: развертывание ключа и шифрование данных. Развертывание ключа преобразует ключ длиной до 448 битов в несколько массивов подключей, общим объемом 4168 байтов.

Шифрование данных состоит из простой функции, последовательно выполняемой 16 раз. Каждый этап состоит из зависимой от ключа перестановки и зависимой от ключа и данных подстановки. Используются только сложения и XOR 32-битовых слов. Единственными дополнительными операциями на каждом этапе являются четыре извлечения данных из индексированного массива.

В Blowfish используется много подключей. Эти подключи должны быть рассчитаны до начала шифрования или дешифрирования данных.

P-массив состоит из 18 32-битовых подключей:

P1, P2, . . ., P18

Каждый из четырех 32-битовых S-блоков содержит 256 элементов:

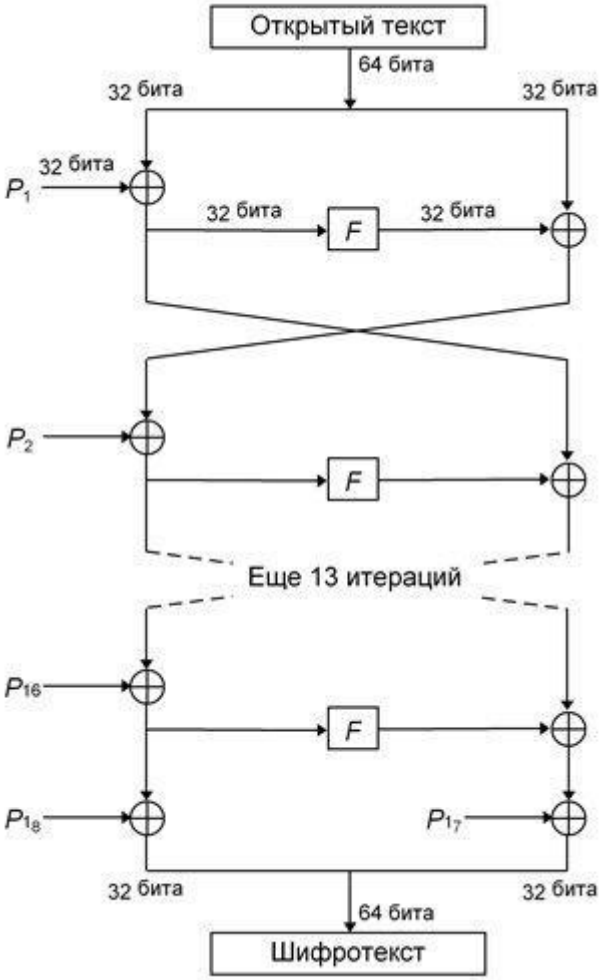
S1,0, S1,1, . . ., S1,255

S2,0, S2,2, . . ., S2,255

S3,0, S3,3, . . ., S3,255

S4,0, S4,4, . . ., S4,255

Рассмотри более подробно вычисление подключей:



На вход подается 64-битовый элемент данных x. Для шифрования разбиваем x на две 32-битовых половины: x_L и x_R

Для этапов с первого по шестнадцатый повторяется (i = 1 по 16):

$x_L = x_L \oplus P_{18}$

$x_R = F(x_L) \oplus x_R$

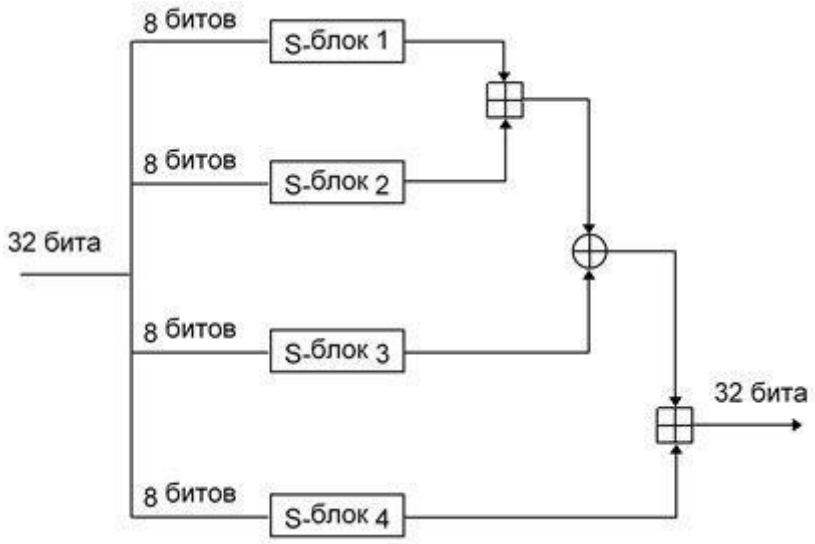
Переставить x_L и x_R (кроме последнего этапа.)

$x_R = x_R \oplus P_{17}$

$x_L = x_L \oplus P_{18}$

Объединить x_L и x_R.

Функция F представляет собой следующее:



Разделить x_L на четыре 8-битовых части: a, b, c и d и выполнить над a, b, c ,d:

$F(x_L) = ((S_{1,a} + S_{2,b}) \oplus S_{3,c}) + S_{4,d}$

Дешифрирование выполняется точно также, как и шифрование, но P1, P2, . . ., P18 используются в обратном порядке.

Подключи рассчитываются с помощью специального алгоритма. Вот какова точная последовательность действий.

1. Сначала P-массив, а затем четыре S-блока по порядку инициализируются фиксированной строкой. Эта строка состоит из шестнадцатиричных цифр p.

2. Выполняется XOR P_1 с первыми 32 битами ключа, XOR P_2 со вторыми 32 битами ключа, и так далее для всех битов ключа (до P_{18}). Используется циклически, пока для всего P-массива не будет выполнена операция XOR с битами ключа.

3. Используя подключи, полученные на этапах (1) и (2), алгоритмом Blowfish шифруется строка из одних нулей.

4. P_1 и P_2 заменяются результатом этапа (3).

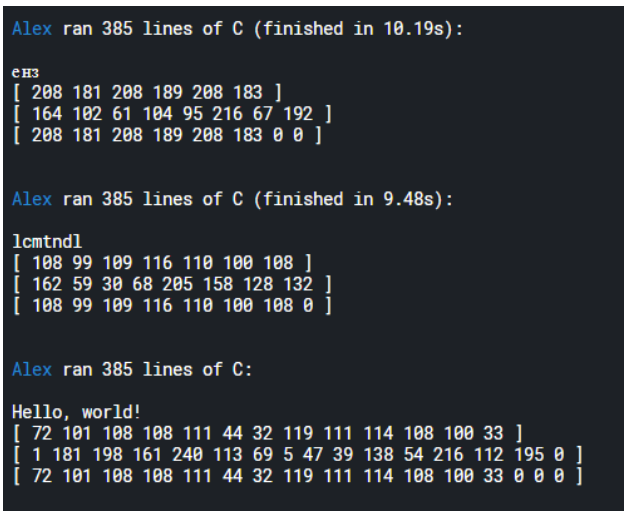
5. Результат этапа (3) шифруется с помощью алгоритма Blowfish и измененных подключей.

6. P_3 и P_4 заменяются результатом этапа (5).

7. Далее в ходе процесса все элементы P-массива и затем по порядку все четыре S-блока заменяются выходом постоянно меняющегося алгоритма Blowfish.

Всего для генерации всех необходимых подключей требуется 521 итерация. Приложения могут сохранять подключи - нет необходимости выполнять процесс их получения многократно.

Результат выполнения программы:



Мы можем видеть, что при шифровании сообщения «Hello, world!» у нас выводится:

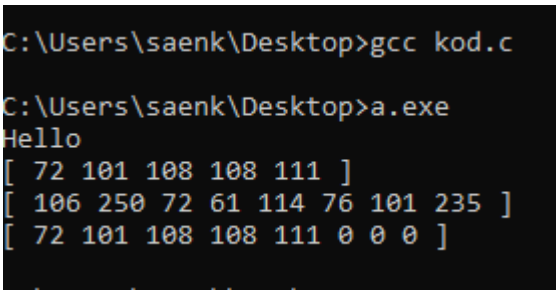
1 строка – текст сообщения в байтах

2 строка – зашифрованное сообщение в байтах

3 строка – расшифрованное сообщение, которое совпадает с первой строкой, что говорит о правильности шифрования. (нули в конце строки – дополнения до блока 64 бит)

И также необходим компилятор, который принимает файл с исходным кодом на Си и компилирует его в исполняемый файл .exe. В качестве компилятора был использован наиболее популярный на сегодня gcc.

Чтобы скомпилировать исходный код, необходимо компилятору gcc передать в качестве параметра файл kod.c, после этого будет скомпилирован исполняемый файл, который в Windows по умолчанию называется a.exe И мы можем обратиться к этому файлу.



Далее подпишем a.exe с помощью команд Windows PowerShell.

Для этого с помощью команды создадим сертификат

New-SelfSignedCertificate -DnsName test2 -Type CodeSigning

Далее экспортировать созданный сертификаты из папки Личные в Доверенные корневые центры сертификации, воспользовавшись консолью certlm.msc.

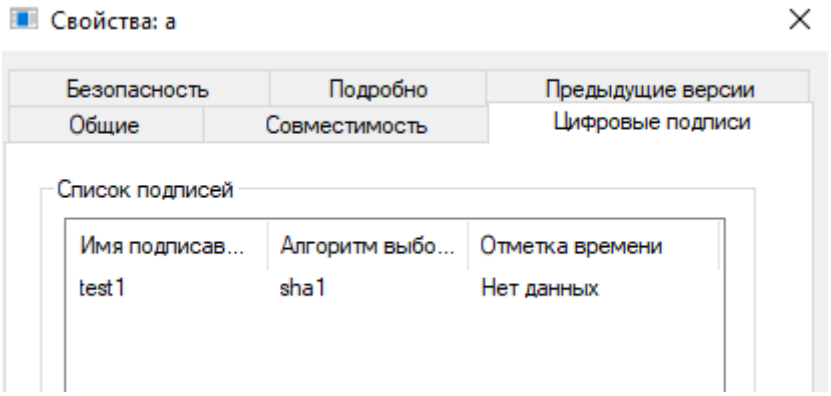
И в заключении подписать файл командой:

Set-AuthenticodeSignature C:\Users\saenk\Desktop\a.exe @(Get-ChildItem cert:\LocalMachine\My -codesigning)[1]

```
PS C:\Windows\system32> Set-AuthenticodeSignature C:\Users\saenk\Desktop\a.exe @(Get-ChildItem cert:\LocalMachine\My -codesigning)[1]

    Каталог: C:\Users\saenk\Desktop

SignerCertificate                         Status                                                                  Path
-----------------                         ------                                                                  ----
818698FAF67753448F8F7BD9D68D03997929C710  Valid                                                                   a.exe
```



Код программы представлен в файле kod.c

Вывод: в ходе выполнения лабораторной работы были освоены отличия и особенности симметричных и ассиметричных алгоритмов шифрования. Детально рассмотрен и изучен алгоритм шифрования Blowfish и реализован на языке программирования Си. Была выполнена процедура генерации ключей, шифрования и дешифрования без использования криптографических библиотек. Программа запускается в среде Windows, исполняемый файл имеет расширение .EXE. Во второй части работы полученный файл a.exe был подписан с помощью команд Windows PowerShell.