**Министерство науки и высшего образования Российской Федерации**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Факультет безопасности информационных технологий**

**Дисциплина:**

«Криптографические методы обеспечения информационной безопасности»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6**

«Модель протокола защищенного соединения»

**Выполнил:**

Молитвин Илья Алексеевич, студент группы N3Изображение выглядит как текст, соединитель

Контент, сгенерированный ИИ, может содержать ошибки.345

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

**Проверил:**

Таранов Сергей Владимирович

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(отметка о выполнении)

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(подпись)

**Оглавление**

[Введение 3](#_Toc162735323)

[Ход работы 4](#_Toc162735324)

[1. Визуализация 1 раунда алгоритма MD5 4](#_Toc162735325)

[2. Алгоритм для генерации кодов аутентификации сообщений HMAC 8](#_Toc162735326)

[3. Хэширования файлов и сообщений с помощью openssl 9](#_Toc162735327)

[3.1. Хэширование сообщения 9](#_Toc162735328)

[3.2. Хэширование файла 10](#_Toc162735329)

[3.3 Генерация Hash based MAC 10](#_Toc162735330)

[4. Криптосистема RSA 10](#_Toc162735331)

[5. Шифрование и дешифрование сообщения с помощью симметричной криптосистемы и openssl 12](#_Toc162735332)

[Вывод 14](#_Toc162735333)

# Введение

Цель работы – изучить подходы к применению криптопримитивов в рамках

протоколов для защищенных соединений.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить взаимодействие таких криптопримитивов как алгоритмы

симметричного и асимметричного шифрования, имитовставки и

электронной подписи.

# Ход работы

## Визуализация 1 раунда алгоритма MD5

В качестве входных данных возьмем стандартный текст:

“Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum a sagittis lectus. In eget leo vel odio molestie ullamcorper. Donec sit amet justo quis nunc varius rhoncus. Morbi et est commodo, sollicitudin odio ut, pretium tellus. Duis rutrum eu mauris nec egestas. Cras maximus mi sem, ac varius elit iaculis vel.”

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 1 - Результат хэширования текста

По стандарту на выходе получаем хэш длиной 128 бит. Рассмотрим подробно, что представляет из себя раундовая функция MD5.

1. Append Padding Bits

В конец входной строки дописываем единичный байт 0х80, а затем нули, до тех пор, пока длина сообщения не будет сравнима с 448 по модулю 512 (то есть, станет на 64 бита короче числа, кратного 512).

1. Append Length

В полученное сообщение дописывается 64-битное представление длины исходного сообщения.

1. Инициализация буфера (Initialize MD Buffer)

Инициализируются 4 переменные (переменные сцепления): A - D, каждая по 32 бита. В них далее будут записываться промежуточные результаты вычислений.

Начальные значения (инициализирующий вектор) представлены на рисунке ниже (они всегда одинаковые):

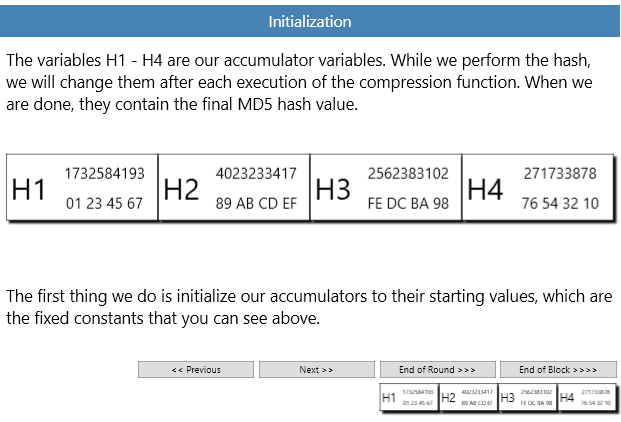


Рисунок 2 - Инициализирующий вектор

1. Чтение данных

Считывание блока данных. Блоки, обрабатываемые функцией сжатия, составляют 64 байта - продолжаем считать до получения 64 байтов.

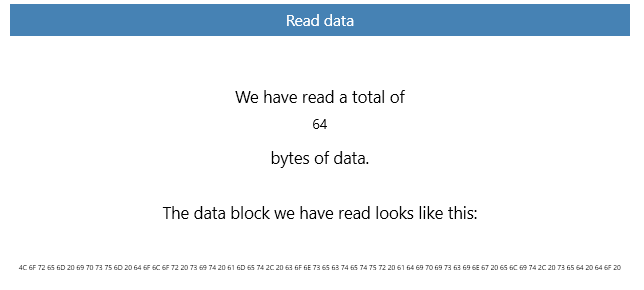


Рисунок 3 - Чтение данных

1. Вычисление

Считанный блок данных делится на 16 32х-битных чисел, так как функция сжатия работает с 32х-битными числами.

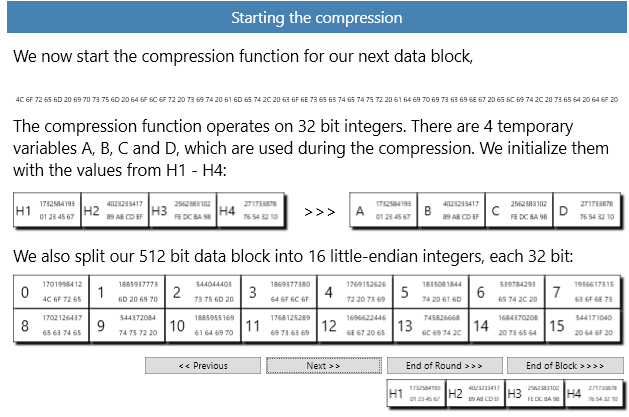


Рисунок 4 - Начало сжатия

Начало сжатия - начало 1 из 4 этапов вычислений, в каждом из которых по 16 шагов.

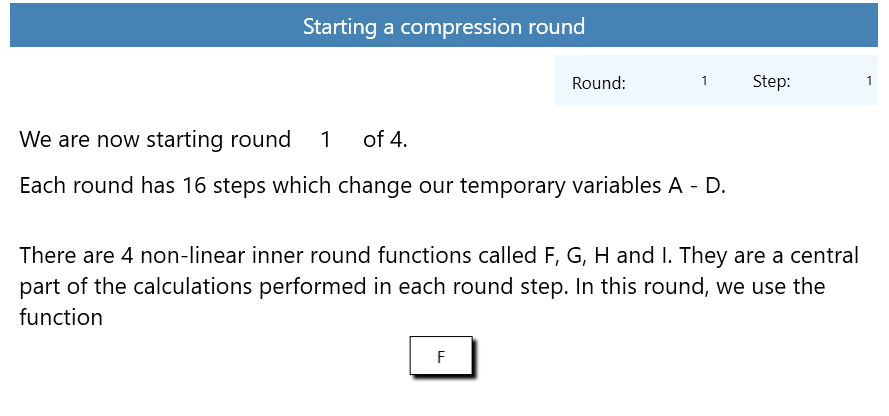


Рисунок 5 - Процесс вычислений

На каждом этапе вычисления одна из этих функций:

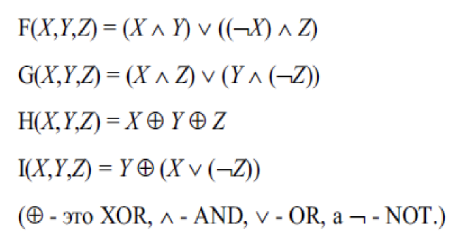


Рисунок 6 - Раундовые функции MD5

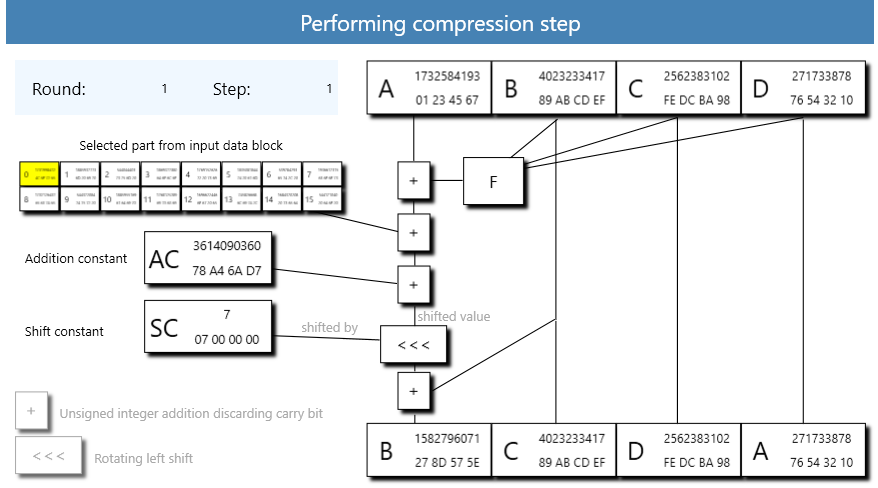


Рисунок 7 - Порядок вычисления

Для блоков B-D выполняются следующие операции:

* сложение результата функции с блоком А по модулю
* результат складывается с блоком исходного сообщения (0-15) по модулю
* к результату прибавляется 32-битная константа AC[i]
* происходит циклический сдвиг влево на 7 бит (SC - константа сдвига)
* результат складывается со значение константы B (также по модулю ) и записывается в B
* записываются значения остальных констант

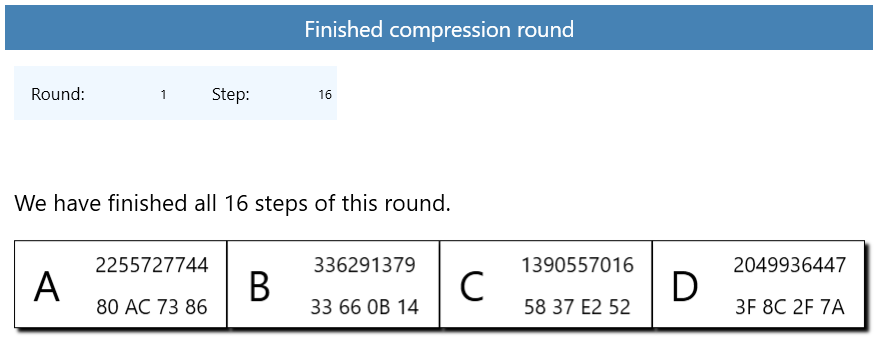


Рисунок 8 – Значения констант по завершении раунда

Полученные в конце раунда значения констант побитово складываются с инициализирующим вектором - результат принимается в качестве начальных значений для следующего раунда.

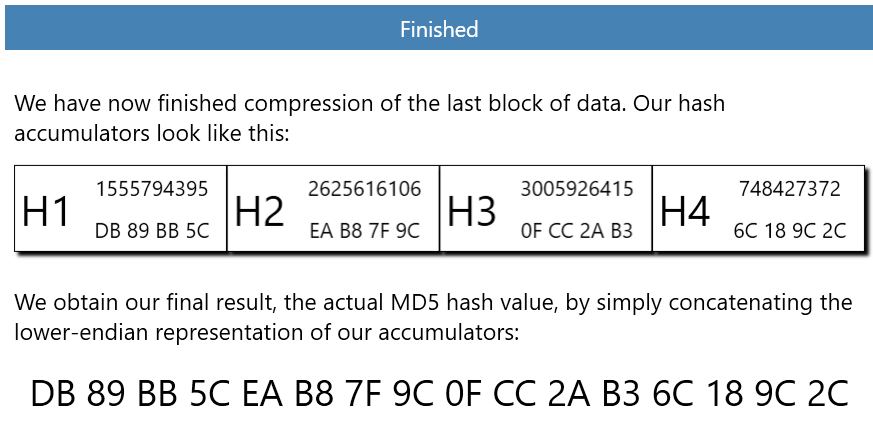


Рисунок 9 – Полученный хэш

## Алгоритм для генерации кодов аутентификации сообщений HMAC

Верхняя половина рабочей области представляет собой набор компонентов, которые вычисляют MD5-HMAC «вручную». Для получения правильного результата ключ HMAC должен быть ровно 64 байта, так как переменные ipad и opad формируются путем XOR-а ключа с соответствующими константами IPAD и OPAD.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 10 – Компоненты “ручного” вычисления MD5-HMAC

Рассмотрим “ручное” вычисление подробнее (по шагам):

1. вычисляем ipad = константа IPAD XOR key (у нас он ровно 64 байта, поэтому просто берем введенные данные) и получаем строку (блок) Si.
2. после конвертирования результата (ipad) производим его конкатенацию (склейку) с исходным сообщением
3. применяем к полученной строке хэш-функцию MD5
4. вычисляем opad = константа OPAD XOR key и получаем строку (блок) So
5. после конвертирования результата (opad) производим его конкатенацию (склейку) результатом шага 3
6. к полученной строке применяем хэш-функцию MD5
7. сравниваем результаты (хэши) “ручного” и “машинного” вычислений

## Хэширования файлов и сообщений с помощью openssl

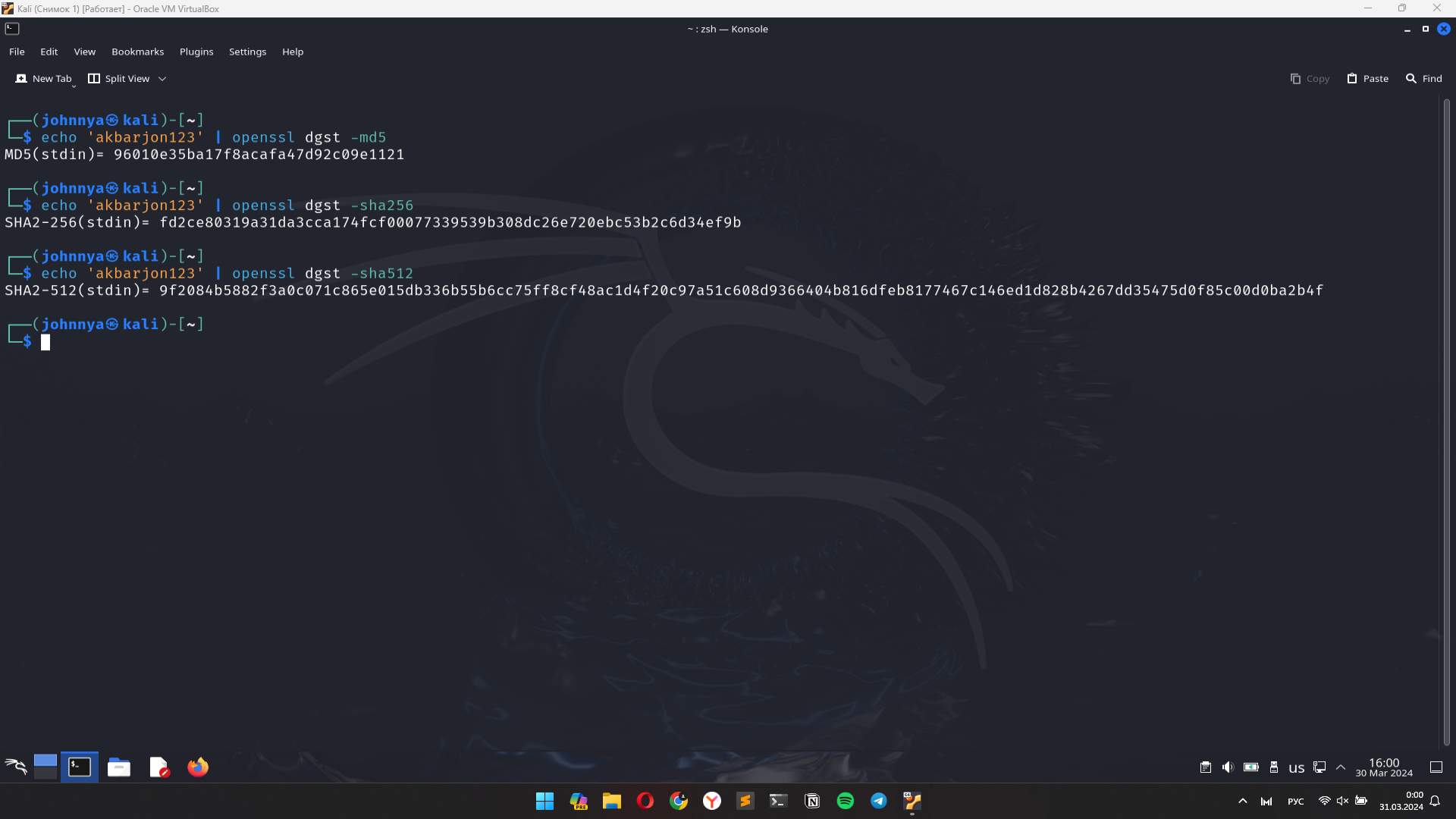
### 3.1. Хэширование сообщения

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 11 – Хэширование сообщения с помощью openssl dgst

Сделаем небольшие изменения во входном тексте.

Рисунок 12 – Хеширование видоизмененного сообщения

### 3.2. Хэширование файла

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 13 – Хеширование файла

### 3.3 Генерация Hash based MAC

Сгенерируем Hash based MAC. Для этого потребуется ключ -hmac с указанием ключа.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 14 – Hash based MAC

## Криптосистема RSA

Сгенерируем парные ключи (открытый и закрытый) для асимметричной криптосистемы RSA.

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описаниеРисунок 15 – RSA

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 16 – RSA

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 17 – RSA

С помощью данной ключевой пары зашифруем случайный ключ AES длиной 128 бит.

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 18 – RSA

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 19 – RSA

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 20 – RSA

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 21 – RSA

## Шифрование и дешифрование сообщения с помощью симметричной криптосистемы и openssl

Возьмем симметричную криптосистему RC4 и зашифруем с ее помощью файл

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 23 –RC4

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 24– RC4

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 25 – RC4

## 

# 

# Вывод

Таким образом, мы познакомились с работой алгоритма MD5, а также с функционалом библиотеки openssl, с помощью которой зашифровали сообщения и файлы, используя алгоритмы SHA-256, SHA-512.