**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 ПРАКТИЧЕСКОЙ РАБОТЕ №6**

«Модель протокола защищенного соединения»

**Выполнил:**

Изображение выглядит как зарисовка, Штриховая графика, Шрифт, рукописный текст

Автоматически созданное описаниеПолевцов Артем Сергеевич, студент группы N34511

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

(подпись)

**Проверил:**

Волков Александр Григорьевич, инженер ФБИТ

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

(отметка о выполнении)

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

(подпись)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc154059588)

[1. МОДЕЛЬ ПРОТОКОЛА ЗАЩИЩЕННОГО СОЕДИНЕНИЯ 4](#_Toc154059589)

[1.1 Ход работы 4](#_Toc154059590)

[1.1.1 Визуализация 1 раунда алгоритма хэширования md5 4](#_Toc154059591)

[1.1.2 Визуализация алгоритма для генерации кодов аутентификации сообщений 6](#_Toc154059592)

[1.1.3 Обзор возможностей хэширования файлов и сообщений с помощью openssl dgst 8](#_Toc154059593)

[1.1.4 Генерация hash based mac 9](#_Toc154059594)

[1.1.5 Формирование ключевой пары rsa, шифрованиесимметричного ключа, генерация и проверка подписи 10](#_Toc154059595)

[1.1.6 Сравнение двух хешей based mac 13](#_Toc154059596)

[ВЫВОД 14](#_Toc154059597)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc154059598)

ВВЕДЕНИЕ

Цель: изучить подходы к применению криптопримитивов в рамках протоколов для защищенных соединений.

Задачи практической работы

1. Выполнить визуализацию 1 раунда алгоритма хэширования MD5;
2. Отразить в отчете алгоритм для генерации кодов аутентификации сообщений;
3. Протестировать возможности хэширования файлов и сообщений с помощью openssl dgst;
4. Генерация Hash based MAC.
5. МОДЕЛЬ ПРОТОКОЛА ЗАЩИЩЕННОГО СОЕДИНЕНИЯ
   1. Ход работы

### Визуализация 1 раунда алгоритма хэширования md5

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 1 – Алгоритм MD5

MD5 (Message Digest Algorithm 5) - это криптографический алгоритм хеширования, который принимает на вход сообщение произвольной длины и выдаёт фиксированный 128-битный хеш-код, обычно представляемый в виде 32-символьной шестнадцатеричной строки.

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Рисунок 2 – Переменные-аккумуляторы

На первом этапе первого раунда происходит инициализация внутреннего состояния (часто обозначаемого как H1-H4 – это переменные-аккумуляторы, меняющиеся после выполнения функции сжатия) предварительными значениями, которые представляют из себя фиксированные константы. Эти значения образуют хеш-значение, которое будет обновляться по мере прохождения данных через алгоритм. И в конце выполнения хеш-функции данные переменные будут содержать ответ.

Далее осуществляется процесс считывания данных. Функция сжатия MD5 обрабатывает данные блоками размером 64 байта. Ее задача заключается в том, чтобы прочитать достаточное количество данных для последующего вызова функции сжатия. Таким образом, она продолжает считывать данные до тех пор, пока не наберется 64 байта.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 3 – Процесс чтения данных

Затем активируется функция сжатия для обработанного блока данных. В процессе сжатия используются четыре временные переменные: A, B, C и D, которые инициализируются значениями H1-H4.

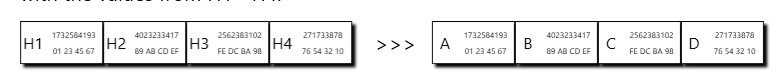


Рисунок 4 – Инициализация переменных

Полученный на вход блок данных делится на 16 блоков, каждый по 32 бита.

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рисунок 5 – Разделение входного потока данных на блоки

Затем начинается выполнение первого раунда сжатия, и общее количество таких раундов составляет 4. Каждый раунд включает в себя 16 шагов, которые влияют на временные переменные A-D. Для этих манипуляций используются нелинейные внутренние раундовые функции с обозначениями F, G, H и I. Эти функции являются ключевой частью вычислений, выполняемых на каждом этапе раунда. В данном случае в первом раунде используется функция F.

Изображение выглядит как диаграмма, текст, снимок экрана, линия

Автоматически созданное описание

Рисунок 6 – Начало первого раунда

И в итоге выполнения 16 схожих шагов мы получаем результат.

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Рисунок 7 – Результат первого раунда

По завершении последнего раунда функции сжатия для получения окончательного результата выполняется операция сложения временных переменных A-D с соответствующими переменными-аккумуляторами.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 8 – Хеш

### Визуализация алгоритма для генерации кодов аутентификации сообщений

HMAC (Hash-based Message Authentication Code) представляет собой криптографическую конструкцию, используемую для создания кода аутентификации сообщения на основе хеш-функции. Эта конструкция обеспечивает сочетание ключа и данных сообщения для создания фиксированной длины кода, который может быть использован для проверки целостности и подлинности сообщения.

Изображение выглядит как текст, снимок экрана, диаграмма, дисплей

Автоматически созданное описание

Рисунок 9 – HMAC в Cryptool 2

HMAC гарантирует участвующим в транзакции сторонам, что их сообщения не были перехвачены злоумышленником. Код (ключ) HMAC состоит из двух частей:

Общий набор криптографических ключей для отправителя (клиента) и получателя (сервера). Отправитель и получатель используют один и тот же ключ для создания и проверки HMAC;

Общая криптографическая хэш-функция, такая как SHA-1 или RIPEMD-128/60.

Формула для HMAC:

HMAC = hashFunc (секретный ключ + сообщение)

В процессе обмена сообщениями между клиентом и сервером с использованием HMAC клиент создает уникальный HMAC (хэш), хэшируя данные запроса с закрытыми ключами и отправляя их как часть запроса. Сервер получает запрос и регенерирует свой собственный уникальный HMAC. Затем он сравнивает два HMAC. Если они равны, клиент считается доверенным, и запрос выполняется. Этот процесс часто называют секретным рукопожатием (secret handshake).

Изображение выглядит как текст, снимок экрана, диаграмма, дисплей

Автоматически созданное описание

Рисунок 10 – HMAC в Cryptool 2

Представим алгоритм HMAC в виде формулы:

HMACK(K, text) = H ((K0 ⊕ opad) ‖ H((K0 ⊕ ipad) ‖ text)), где:

− «⊕» – исключающее «ИЛИ»;

− «‖» – склейка строк.

H - хэш-функция пароля (такая как MD5 или SHA-2) , который может выполнять

сжатие группового цикла;

K — ключ (секретный ключ);

text — сообщение, которое должно быть аутентифицировано;

K0 — другой ключ, полученный из исходного ключа K (если K короче размера входного блока хэша). функция, затем правое дополнение нулями; решетка K, если она больше размера блока)

внутреннее заполнение ipad (0x5C5C5C...5C5C, шестнадцатеричная константа);

внешнее заполнение opad (0x363636...3636, шестнадцатеричная константа)

### Обзор возможностей хэширования файлов и сообщений с помощью openssl dgst

Захешируем два произвольных сообщения, различающихся только регистром первого символа при помощи openssl dgst, применяя алгоритм md5

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 11 – Хеширование сообщения при помощи Openssl

Видим два хеша длиной 128 бит совершенно не похожих друг на друга. Что говорит о том, что минимальные изменения в исходном сообщении кардинально меняют внешний вид хеша.

Теперь протестируем другие алгоритмы хеширования – SHA-256 и SHA-512.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 12 - Хеширование сообщения при помощи Openssl

Видим, что выходные значения не похожи и имеют разную длину.

### Генерация hash based mac

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 13 – Вычисление HMAC-MD5

Из рисунка выше мы видим, что вычисления HMAC-MD5прошли успешно и все выходные значения отличаются друг от друга, поскольку каждый раз был использован другой пароль.

Теперь вычислим HMAC-SHA-256 и HMAC-SHA-512.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 14 – Вычисление HMAC-SHA-256 и HMAC-SHA-512

Как можем видеть, вычисления прошли успешно.

### Формирование ключевой пары rsa, шифрованиесимметричного ключа, генерация и проверка подписи

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Рисунок 15 – Генерация ключа RSA

В этом файле помимо закрытого ключа также содержится и открытый.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 16 – Публичный ключ

Выше мы наблюдаем публичный ключ в отельном файле.

Затем нам нужно сгенерировать симметричный ключ системы AES и поместить его в файл aes\_key.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 17 – Генерация симметричного ключа

Далее выполним шифрование симметричного ключа при помощи открытого.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 18 – Симметричный ключ

Расшифруем при помощи закрытого ключа симметричный ключ.

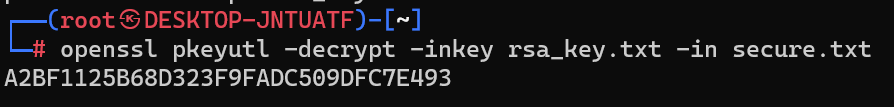


Рисунок 19 – Расшифрованный симметричный ключ

Генерируем хеш для зашифрованного ключа.

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Автоматически созданное описание

Рисунок 20 – Генерация хеша зашифрованного симметричного ключа

Подписываем получившийся хеш при помощи сгенерированного ключа RSA.

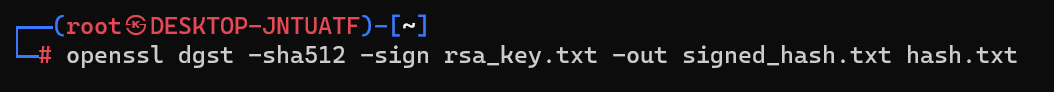


Рисунок 21 – Подпись хеша

Проверяем подписанный хеш.

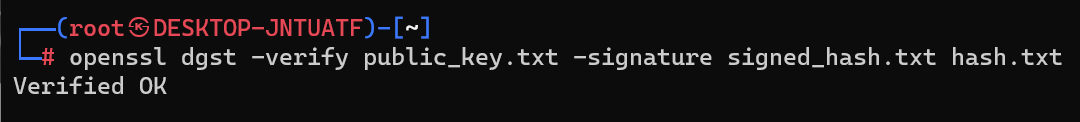


Рисунок 22 – Проверка подписи хеша

Продемонстрируем процесс шифрования и дешифрования открытого сообщения прим помощи симметричной криптосистемы на примере AES, при помощи команды openssl enc.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 23 – Шифрование и дешифрование исходного сообщения

### Сравнение двух хешей based mac

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 24 – Сравнение двух HMAC

Для того, чтобы окончательно убедиться в корректности. Создадим еще один HMAC для первого открытого сообщения.

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Автоматически созданное описание

Рисунок 25 – Сравнение идентичных HMAC

ВЫВОД

В ходе данной лабораторной работы были рассмотрены алгоритмы хеширования MD5, SHA-1, SHA256, SHA512. Были сгенерированы пары открытый-закрытый ключи для криптосистемы RSA. Был зашифрован случайный симметричный ключ алгоритма AES. Далее было проведено сравнение двух идентичных HMAC с помощью команды diff для операционных систем linux.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бабенко, Л. К. Современные алгоритмы блочного шифрования и методы их анализа / Л.К. Бабенко, Е.А. Ищукова. - М.: Гелиос АРВ, 2015. - 376 c.
2. Бабенко, Л.К. Современные интеллектуальные пластиковые карты / Л.К. Бабенко. - М.: Гелиос АРВ, 2015. - 921 c.
3. Болотов, А. А. Элементарное введение в эллиптическую криптографию. Протоколы криптографии на эллиптических кривых / А.А. Болотов, С.Б. Гашков, А.Б. Фролов. - М.: КомКнига, 2012. - 306 c.
4. Бузов, Геннадий Алексеевич Защита информации ограниченного доступа от утечки по техническим каналам / Бузов Геннадий Алексеевич. - М.: Горячая линия - Телеком, 2016. - 186 c.