МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Самарский национальный исследовательский университет

имени академика С. П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Факультет информатики

Кафедра программных систем

ОТЧЁТ

по лабораторной работе № 4

«Криптографические методы защиты информации»

по курсу «Информационная безопасность корпоративных систем»

Выполнил:

Лазарев М.Ю.

Дьяконов А.В.

гр. 6132-020402D

Проверил:

Додонов М. В.

Самара 2024

Цель работы: Изучение базовых криптографических операций.

Ход выполнения лабораторной работы.

**Задание 1. Кодирование Base64.**

1. Запускаем программу cryptography\_study соответствующим exe-файлом. Выбираем «Преобразование в Base64». Вводим соответствующий текст, переключим кодировку источника ASCII/UTF-8 (Рис. 1). В стандарт ASCII не входят кириллические символы, поэтому кириллический текст не может быть представлен. В UTF-8 поддерживает все символы Юникода, включая кириллические и каждый кириллический символ кодируется с использованием нескольких байтов (обычно 2 байта для большинства символов). Когда мы вводим кириллические символы и кодируем их в Base64, UTF-8 использует больше байтов для представления каждого символа по сравнению с латинскими символами. Это приводит к увеличению размера закодированного сообщения Base64.

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

Автоматически созданное описаниеИзображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 1 – Кодировка ASCII и UTF-8, преобразованная в Base64

1. Добавляем в источник латинские символы, пока символ «=» в результате не исчезнет. На каждый третий одинаковый символ наблюдаем, что «=» в конце исчезает. Это связано с тем, что Base64 кодирует каждые 3 байта исходных данных в 4 символа Base64. Если количество байтов в исходных данных кратно 3, то символы «=» не добавляются, так как выравнивание не требуется. Base64 кодирует данные, разбивая их на группы по 3 байта (24 бита) и затем преобразуя эти 24 бита в 4 символа Base64 (каждый символ представляет 6 бит). Если количество байтов в исходных данных не кратно 3, то для завершения последней группы добавляются нулевые биты. Чтобы указать, что эти дополнительные биты не являются частью исходных данных, в конце закодированного сообщения добавляются один или два символа «=».

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

Автоматически созданное описаниеИзображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 2 – Добавление латинского символа

1. Копируем изначальный результат из «Преобразовать в Base64» в источник и выбираем «Преобразовать из Base64». (Рис. 3)

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

Автоматически созданное описание

Рисунок 3 – Преобразование из Base64

1. Результат как ASCII не меняет сообщение. При выборе результата как UTF-8 наблюдаем артефакты на месте кириллицы (Рис. 4).

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

Автоматически созданное описание

Рисунок 4 – Результат как UTF-8

1. Повторяем п.3 и п.4, но указываем в источнике «Текст UTF-8», а в «Преобразовать в Base64» - «Результат как UTF-8» (Рис. 5)(Рис. 6). При переключении результата на «Результат как ASCII», также наблюдаем артефакты (Рис. 7).

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

Автоматически созданное описание

Рисунок 5 – «Результат как UTF-8»

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

Автоматически созданное описание

Рисунок 6 – «Преобразовать из Base64»

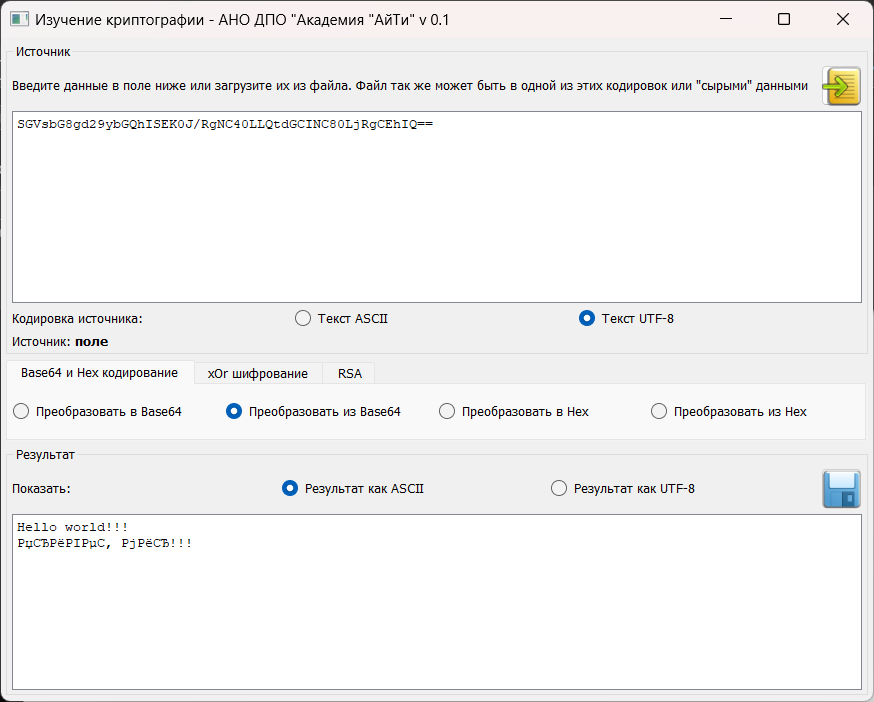


Рисунок 7 – «Результат как ASCII»

1. По результатам можно однозначно сказать, что Base64 не изменяет исходную кодировку текста, а лишь преобразует данные в текстовый формат.

При кодировании Base64 исходные данные (независимо от их кодировки) разбиваются на группы по 3 байта и затем преобразуются в 4 символа Base64. Этот процесс не изменяет исходную кодировку данных, а просто преобразует их в другой формат. При декодировании Base64 данные преобразуются обратно в их исходный бинарный формат. Если исходные данные были в кодировке UTF-8, они останутся в UTF-8 после декодирования. Поэтому появляются артефакты при переключении кодировки.

**Задание 2. Похищение закрытого ключа шифрования с помощью кодирования Base64.**

1. Создаем текстовый файл с текстом EFS.txt (Рис. 8). Открываем «Свойства», вкладка «Общее». Выбираем «Другие…». В окне «Дополнительные атрибуты» отмечаем «Шифровать содержимое для защиты данных», нажимаем «Ок». Применяем параметры и в окне «Предупреждение о шифровании» выбираем «Зашифровать только файл» (Рис. 9). Теперь данный файл защищен EFS шифрованием и создан сертификат администратора с закрытым ключем.

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

Автоматически созданное описание

Рисунок 8 – Свойства файла

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9 - Шифрование

1. Нажимаем комбинацию Win + R и в поле «Открыть» вводим certmgr.msc (Рис. 10). В открывшемся окне, в ветке «Сертификаты – текущий пользователь» открываем «Личное» и кликаем на «Сертификаты». (Рис. 11)

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 10 – Окно «Выполнить»

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 11 – Окно сертификатов

1. Экспортируем сертификат с закрытым ключом. Для этого кликаем правой мышкой по сертификату и выбираем «Все задачи» - «Экспорт» (Рис. 12). Открывается окно «Мастер экспорта сертификата», в окне выбираем «Да, экспортировать закрытый ключ» - «Далее», в окне выбора формата «Далее», в опциях безопасности ставим пароль «1», в выборе имени – «Обзор» (сохраняем в любую нужную нам директорию) – «Далее» (Рис. 13). В последнем окне кликаем «Готово» (Рис. 14).

Изображение выглядит как текст, программное обеспечение, Значок на компьютере, веб-страница

Автоматически созданное описание

Рисунок 12 – Экспортируем сертификат

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеИзображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описаниеИзображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 13 – «Мастер экспорта сертификата»

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

Автоматически созданное описание

Рисунок 14 – Завершение экспорта сертификата

1. Запускаем cryptography\_study и загружаем из получившегося файла как «сырые данные» сохраненный PFX файл с преобразованием в Base64 (Рис. 15) (Рис. 16). Сохраняем в текстовый файл результаты (Рис. 17) и получаем TXT файл, который можно забрать с собой на любом носителе (Рис. 18).

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

Автоматически созданное описание

Рисунок 15 – Открытие файлаИзображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 16 – Преобразование в Base64

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

Автоматически созданное описание

Рисунок 17 – Сохранение результата

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

Автоматически созданное описание

Рисунок 18 – Готовый текстовый файл

**Задание 3. Кодирование Hex (шестнадцатеричное).**

1. Запускаем cryptography\_study. Выбираем тип кодирования «Преобразование в Hex». При переключении кодировки источника ASCII/UTF-8 меняется только часть результата соответствующая кириллице. (Рис. 19)

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

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 19 – Смена кодировки источника

ASCII:

0010 cf f0 e8 e2 e5 f2 20 ec e8 f0 21 21 21

UTF-8:

0010 d0 9f d1 80 d0 b8 d0 b2 d0 b5 d1 82 20 d0 bc d0

0020 b8 d1 80 21 21 21

1. Копируем полученные результаты в источник и выбираем «Преобразовать из Hex» для обратного кодирования. (Рис. 20)

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

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 20 – Обратное кодирование

**Задание 4. Изучение операции XOR.**

1. Запускаем cryptography\_study, выбираем вкладку «xOr шифрование» и отмечаем «Ключ из файла». (Рис. 21)

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

Автоматически созданное описание

Рисунок 21 – Окно программы

1. Выбираем файл \cryptography\_study\key.txt, нажав на соответствующую иконку и получаем результат. Сохраняем результат в DAT файл. Открываем сохраненный файл в программе, нажав «Загрузить как ASCII…». Убеждаемся в корректности результата. (Рис. 22)

.Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 22 – До и после произведенных манипуляций

1. Меняем в key.txt ключ шифрования (Рис. 23). И видим только часть расшифрованного текста (Рис. 24). Часть расшифровки корректна, а часть - нет, так как ключ шифрования не полный. В отличие от ключа, который использовался изначально.

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

Автоматически созданное описание

Рисунок 23 – Изменённый файл key.txt

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

Автоматически созданное описание

Рисунок 24 – Изменённый результат

**Задание 5. Изучение шифра гаммирования на операции XOR.**

1. Запускаем cryptography\_study, выбираем вкладку «xOr шифрование» и отмечаем «Гаммирование». Указываем начальное значение генератора гаммы – 2 (Рис. 25). Сохраняем результат в файл crypt.dat.

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

Автоматически созданное описание

Рисунок 25 – Гаммирование

1. Загружаем в источник сохраненный файл. При значении гаммирования – 2, результаты корректны. При смене значения генератора гаммы соответственно результаты становятся некорректными. (Рис. 26)

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

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 26 – Корректный и некорректные показания