УО «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра «Информационных систем и технологий»

Специальность 1-40 05 01 «Информационные Системы и Технологии»

**Основы информационной безопасности**

**Практическое задание № 5**

**Вариант 17**

|  |  |  |  |
| --- | --- | --- | --- |
| **Выполнил** |  |  |  |
| Студент 2 курса группы 1 |  |  | Д.И. Велютич |
|  | подпись, дата |  | инициалы и фамилия |
| **Проверил(а)** |  |  | Н.В. Ржеутская |
|  | подпись, дата |  | инициалы и фамилия |

Минск 2023

**Тема «Криптографическая защита информации»**

Цель: изучить и закрепить умение реализации ЭЦП на примере RSA**.**

**Теория**

ЭЦП RSA - это один из наиболее распространенных методов создания цифровых подписей. Эти протоколы обеспечивают гарантию того, что сообщение было отправлено от достоверного отправителя и не было изменено в процессе передачи. Получатель может использовать ЭЦП как доказательство достоверности сообщения перед третьими лицами.

Для создания цифровой подписи отправитель применяет к исходному сообщению хеш-функцию, которая преобразует его в фиксированный набор битов. Затем отправитель вычисляет цифровую подпись с использованием своего закрытого ключа RSA. Получатель проверяет соответствие хеш-образов и если они совпадают, то сообщение пришло от достоверного отправителя и не было искажено.

Кроме классической схемы ЭЦП RSA существуют другие разновидности, такие как "конфиденциальная" подпись, "вслепую" подпись, "мультиподпись" и другие. Каждый тип подписи имеет свои особенности и предназначен для определенных задач.

Процесс создания ЭЦП RSA проходит в несколько этапов. В первом этапе отправитель вырабатывает свои открытый и закрытый ключи RSA. Второй этап состоит в отправке сообщения и цифровой подписи. Наконец, на третьем этапе получатель получает сообщение и проверяет электронную подпись.

ЭЦП RSA широко используется в различных областях, таких как электронная коммерция, интернет-банкинг, цифровые документы и многие другие. Он обеспечивает высокий уровень безопасности и гарантирует, что сообщение было отправлено от достоверного отправителя и не было изменено в процессе передачи.

**1**) При генерации ЭЦП компьютер отправителя использует закрытый ключ RSA для создания уникальной цифровой подписи для сообщения. При проверке ЭЦП получатель использует открытый ключ RSA отправителя для проверки подписи и убеждается, что сообщение было отправлено именно отправителем и не было изменено.

**2**) При отправке ЭЦП отправитель генерирует цифровую подпись с помощью своего закрытого ключа RSA и отправляет сообщение вместе с этой подписью получателю. Получатель затем использует открытый ключ RSA отправителя для проверки подписи и убеждается, что сообщение было отправлено именно отправителем и не было изменено.

**3**) Схема протокола ЭЦП на основе алгоритма RSA состоит из следующих шагов:

- Генерация открытого и закрытого ключей RSA отправителя.

- Создание цифровой подписи отправителем с помощью его закрытого ключа RSA.

- Отправка сообщения и цифровой подписи получателю.

- Получатель использует открытый ключ RSA отправителя для проверки подписи и убеждается, что сообщение было отправлено именно отправителем и не было изменено.

- Если подпись верна, то сообщение считается достоверным и получатель может принять дальнейшие действия. Если подпись неверна, то сообщение может быть отклонено.

**Условия задачи**

1. Объясните последовательность выполнения процедур генерации и проверки ЭЦП.
2. Опишите последовательность действий участников протокола при отправке и проверке ЭЦП.
3. Опишите схему протокола ЭЦП на основе алгоритма RSA.
4. \*На базе алгоритма RSA получить ЭЦП (в проекте можно использовать существующие криптографические алгоритмы). Удостовериться, что ЭЦП принадлежит именно этому сообщению.

**Исполнительная часть**

from Crypto.PublicKey import RSA

from Crypto.Signature import PKCS1\_v1\_5

from Crypto.Hash import SHA256

# Исходное сообщение

message = b"Hello, world!"

print("Исходное сообщение:", message.decode())

# Генерация ключей RSA

key = RSA.generate(2048)

private\_key = key.export\_key()

public\_key = key.publickey().export\_key()

print("Закрытый ключ:", private\_key.decode())

print("Открытый ключ:", public\_key.decode())

# Создание хеша сообщения

hash = SHA256.new(message)

# Создание подписи с помощью закрытого ключа RSA

signer = PKCS1\_v1\_5.new(key)

signature = signer.sign(hash)

print("Подпись файла:", signature.hex())

# Проверка подписи с помощью открытого ключа RSA

verifier = PKCS1\_v1\_5.new(key.publickey())

if verifier.verify(hash, signature):

    print("Подпись верна")

else:

    print("Подпись неверна")

**Вывод:** После выполнения шагов по изучению принципов работы алгоритма RSA для ЭЦП, изучению необходимых модулей и написанию кода на Python для генерации ключей, создания и проверки подписи, я получил глубокое понимание применения RSA в защите данных. Тестирование кода на различных сообщениях помогло закрепить практический навык реализации ЭЦП на примере RSA. В результате, я уверен в своей способности создавать и проверять подписи при использовании данного алгоритма.