Криптография и сигурност

Лектор: Георги Пашев

Дата: 29.09.2023

Цели на презентацията

•Да запознаем слушателите с основните понятия и алгоритми за кодиране, декодиране, криптиране, декриптиране, хеширане, подписване и проверка на данни
•Да обясним разликата между симетрично и асиметрично криптиране и да представим някои от най-използваните методи за тях
•Да покажем как криптографията се прилага в различни области, като крипто валути, умни договори, дигитални подписи и др.

План на презентацията

•Основни понятия и алгоритми за кодиране, декодиране, криптиране, декриптиране, хеширане, подписване и проверка на данни
•Симетрично и асиметрично криптиране
•Приложения на криптографията в различни области
•Заключение и въпроси

Основни понятия и алгоритми за кодиране, декодиране, криптиране, декриптиране, хеширане, подписване и проверка на данни

•Кодиране е процесът на преобразуване на информация от един формат в друг, като се използва определена правила или схеми. Целта на кодирането е да се оптимизира или стандартизира информацията за по-лесно съхранение, предаване или обработване. Примери за кодиране са ASCII, Unicode, Base64 и др.

•Декодиране е процесът на възстановяване на оригиналната информация от кодираната форма, като се използват същите или обратни правила или схеми.

Целта на декодирането е да се възстанови информацията в разбираем или полезен формат. Примери за декодиране са ASCII, Unicode, Base64 и др.

•Криптиране е процесът на преобразуване на информация в нечетлива или скрита форма, като се използва определен алгоритъм и ключ. Целта на криптирането е да се защити информацията от неоторизиран достъп или модификация. Примери за криптиране са AES, RSA, DES и др.

•Декриптиране е процесът на възстановяване на оригиналната информация от криптираната форма, като се използва същият или обратен алгоритъм и ключ. Целта на декриптирането е да се разкрие информацията на упълномощени лица или системи. Примери за декриптиране са AES, RSA, DES и др.

•Хеширане е процесът на преобразуване на информация в уникална и фиксирана дължина на изход, като се използва определена хеш функция. Целта на хеширането е да се идентифицира или провери информацията по ефективен и безопасен начин. Примери за хеширане са SHA-256, MD5, CRC и др.

•Подписване е процесът на добавяне на допълнителна информация към оригиналната информация, като се използва определен алгоритъм и ключ. Целта на подписването е да се потвърди автентичността, целостта или неотрицаемостта на информацията. Примери за подписване са RSA, DSA, ECDSA и др.

•Проверка е процесът на проверка на допълнителната информация, която е добавена към оригиналната информация, като се използва същият или обратен алгоритъм и ключ. Целта на проверката е да се установи валидността, целостта или неотрицаемостта на информацията. Примери за проверка са RSA, DSA, ECDSA и др.

Симетрично и асиметрично криптиране

•Симетрично криптиране е метод за криптиране, при който се използва един и същи ключ за криптиране и декриптиране на информацията. Симетричното криптиране е бързо, ефективно и сигурно, но изисква предварително споделяне и съхранение на ключа между комуникиращите страни. Примери за симетрични алгоритми са AES, DES, Blowfish и др.
•Асиметрично криптиране е метод за криптиране, при който се използват два различни ключа за криптиране и декриптиране на информацията. Единият ключ е

публичен и може да се споделя свободно, а другият е частен и трябва да се пази таен. Асиметричното криптиране е по-бавно, по-сложно и по-надеждно, но не изисква предварително споделяне и съхранение на ключа между комуникиращите страни. Примери за асиметрични алгоритми са RSA, ECC, ElGamal и др.

Сравнение между симетрично и асиметрично криптиране

•В тази таблица са представени някои от основните разлики между симетрично и асиметрично криптиране по различни критерии, като брой ключове, скорост, сигурност и приложения

Критерий	**Симетрично криптиране**	**Асиметрично криптиране**
Брой ключове	Един ключ	Два ключа
Скорост	Бързо	Бавно
Сигурност	Висока, но зависи от споделянето и съхранението на ключа	Висока, но зависи от дължината и сложността на ключа
Приложения	Шифроване на големи обеми от данни, като файлове, съобщения, видео и др.	Шифроване на малки обеми от данни, като ключове, пароли, подписи и др.

Пример за симетрично криптиране

•В този пример ще видим как можем да използваме симетричен алгоритъм за криптиране и декриптиране на съобщение, като използваме един и същи ключ

•Нека да използваме алгоритъма AES, който е един от най-популярните и сигурни симетрични алгоритми, който работи с блокове от 128, 192 или 256 бита

•Нека да използваме Python библиотеката Cryptography, която предоставя лесен и безопасен начин за работа с криптографски алгоритми

•Нека да използваме ключ с дължина 256 бита, който е **b'\x0c\x9d\x9a\x9b\xcb\x88\x0d\x11\x18\xbe\x86\x08\x81\x31\x9c\x69\x23\x08\xcc\x81\x1a\x94\xcf\xef\x0b\x24\xed\x13\x71\x2c'**

•Нека да използваме съобщение, което е **Hello, world!**

•Нека да създадем обект от тип **Fernet**, който ще използва ключа, който сме избрали, за да криптира и декриптира съобщението

Python

```
from cryptography.fernet import Fernet
key = b'\x0c\x9d\x9a\x9b\xcb\x88\x0d\x11\x18\xbe\x86\x08\x81\x31\x9c\x69\x23\x08\
xcc\x81\x1a\x94\xcf\xef\x0b\x24\xed\x13\x71\x2c'
fernet = Fernet (key)
```

•Нека да криптираме съобщението, като го преобразуваме в байтове и извикаме метода **encrypt** на обекта **fernet**

Python

```
message = "Hello, world!"
encrypted = fernet.encrypt (message.encode ())
print (encrypted)
```

•Резултатът от криптирането е **b'gAAAAABhZmG5pW6S8TjzKV0kx4qtnjLw-8LXMO8TZWfzFZUf8xjG8W9vLH2QsGN_BZ4GzCMVkL8uXQfxnCvVxwQYwzLIXlZ8zw==**'

•Нека да декриптираме резултата, като извикаме метода **decrypt** на обекта **fernet** и го преобразуваме в текст

Python

```
decrypted = fernet.decrypt (encrypted).decode ()
print (decrypted)
```

•Резултатът от декриптирането е **Hello, world!**

Пример за асиметрично криптиране

•В този пример ще видим как можем да използваме асиметричен алгоритъм за криптиране и декриптиране на съобщение, като използваме два различни ключа

•Нека да използваме алгоритъма RSA, който е един от най-популярните и сигурни асиметрични алгоритми, който работи с публичен и частен ключ, които са свързани с математическа връзка

•Нека да използваме Python библиотеката [Cryptography], която предоставя лесен и безопасен начин за работа с криптографски алгоритми

•Нека да създадем публичен и частен ключ с дължина 2048 бита, като използваме функцията **generate_private_key** от модула **cryptography.hazmat.primitives.asymmetric.rsa**

Python

```
from cryptography.hazmat.primitives.asymmetric import rsa
private_key = rsa.generate_private_key (
    public_exponent = 65537,
    key_size = 2048,
)
public_key = private_key.public_key ()
```

•Нека да използваме съобщение, което е **Hello, world!**

•Нека да криптираме съобщението, като го преобразуваме в байтове и извикаме метода **encrypt** на обекта **public_key** с подходящи параметри за падинг

Python

```
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import padding
message = "Hello, world!"
encrypted = public_key.encrypt (
    message.encode (),
    padding.OAEP (
        mgf = padding.MGF1 (algorithm = hashes.SHA256 ()),
        algorithm = hashes.SHA256 (),
        label = None
    )
)
```

•Нека да декриптираме резултата, като извикаме метода **decrypt** на обекта **private_key** и го преобразуваме в текст

Python

```
decrypted = private_key.decrypt (
    encrypted,
```

```
    padding.OAEP (
        mgf = padding.MGF1 (algorithm = hashes.SHA256 ()),
        algorithm = hashes.SHA256 (),
        label = None
    )
).decode ()
print (decrypted)
```

•Резултатът от декриптирането е **Hello, world!**

Приложения на криптографията в различни области

•Криптографията се прилага в множество области, където е необходимо да се защити, провери или идентифицира информация или комуникация. Някои от тези области са:

•Крипто валути и умни договори - криптографията се използва за да се гарантира сигурността, автентичността и неизменността на транзакциите, събитията и данните, които се записват в блокчейн системи. Криптографията също така позволява създаването и изпълнението на умни договори, които са компютърни програми, които автоматично изпълняват предварително дефинирани условия и действия. Примери за крипто валути и умни договори са Биткойн, Етериум, Кардано и др.

•Дигитални подписи и сертификати - криптографията се използва за да се потвърди автентичността, целостта или неотрицаемостта на документи, съобщения или други данни, като се добавя допълнителна информация, която е свързана с източника или съдържанието на данните. Криптографията също така позволява създаването и управлението на дигитални сертификати, които са документи, които удостоверяват идентичността или правомощията на лица или системи. Примери за дигитални подписи и сертификати са RSA, DSA, SSL и др.

•Шифроване на данни и комуникация - криптографията се използва за да се защитят данните и комуникацията от неоторизиран достъп или модификация, като се преобразуват в нечетлива или скрита форма. Криптографията също така позволява разкриването на данните и комуникацията само на упълномощени лица или системи, като се възстановят в оригиналната си форма. Примери за шифроване на данни и комуникация са AES, RSA, PGP и др.

Заключение

•В тази презентация сме разгледали основните понятия и алгоритми за кодиране, декодиране, криптиране, декриптиране, хеширане, подписване и проверка на данни

•Също така сме сравнили разликата между симетрично и асиметрично криптиране и сме представили някои от най-използваните методи за тях

•Освен това сме показали как криптографията се прилага в различни области, като крипто валути, умни договори, дигитални подписи и др.

•Надяваме се, че сме ви заинтригували и информирали за тази интересна и важна тема