Hội thảo 5. Chữ ký số với đường cong elip

Nghiên cứu các phép toán trong nhóm điểm của đường cong elip cho trường hợp liên tục và rời rạc:

https://andrea.corbellini.name/ecc/interactive/reals-add.html

---

mô-đun ecdsa

Trong Python, thuật toán ECDSA được triển khai trong gói ecdsa.

Gói này cung cấp khả năng tạo khóa, ký, xác minh và lấy bí mật chung cho năm đường cong elip phổ biến nhất với độ dài khóa là 192, 224, 256, 384 và 521 bit, bao gồm. đường cong secp256k1 được sử dụng trong Bitcoin và các loại tiền điện tử khác.

Cài đặt gói:

cài đặt pip ecdsa

Gói này định nghĩa các lớp SigningKey và VerifyingKey , tương ứng đại diện cho khóa riêng và khóa chung của thuật toán ECDSA.

Khóa được tạo bằng phương thức SigningKey.generate(curve),
tham số là đường cong được sử dụng trong thuật toán.

Khóa có thể được lưu vào chuỗi byte bằng phương thức to_pem() và đọc từ chuỗi byte bằng phương thức from_pem().

Việc ký tin nhắn bằng khóa riêng được thực hiện bằng phương thức sign(msg) của đối tượng SigningKey, trong đó msg là thông điệp được ký. Kết quả của phương pháp là chữ ký điện tử ở dạng chuỗi byte.

Chữ ký được xác minh bằng phương thức verify(sig, msg) của đối tượng VerifyingKey , trong đó sig là chữ ký điện tử, msg là thông báo đang được xác minh. Nếu kiểm tra thành công, giá trị True sẽ được trả về; nếu không, ngoại lệ BadSignatureError sẽ được tạo.

Bài tập

1. Tạo cặp khóa secp256k1 và lưu vào đĩa.
2. Tải khóa riêng từ đĩa, ký vào tin nhắn và ghi lại
   chữ ký nhận được trên đĩa.
3. Tải khóa chung, tin nhắn và chữ ký từ đĩa. Thực hiện xác minh chữ ký.

Tạo địa chỉ bitcoin

Địa chỉ Bitcoin được tạo dựa trên khóa công khai ECDSA bằng cách sử dụng như sau thuật toán:

1. Một số 32 byte ngẫu nhiên từ 1 đến 0xFFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFE BAAE DCE6 AF48 được chọn làm khóa riêng của thuật toán ECDSA
A03B BFD2 5E8C D036 4141.

2. Khóa chung tương ứng được tạo bằng cách triển khai secp256k1 của thuật toán ECDSA.

Khóa công khai được viết ở dạng không nén, dưới dạng hai khóa 32- liên tiếp số byte đứng trước một byte có giá trị 0x04 - dấu hiệu không nén khóa công khai.

3. Hàm băm SHA-256 của khóa chung được tính toán.

4. Giá trị băm RIPEMD-160 được tính từ giá trị băm của bước trước đó.

Thuật toán RIPEMD-160 hiện được coi là lỗi thời. Sự hỗ trợ của anh ấy trong thư viện OpenSSL 3.0 đã ngừng hoạt động và chức năng tương ứng trong mô-đun hashlib bị chặn. Để sử dụng nó trong các chương trình, bạn có thể tải xuống và cài đặt một mô-đun riêng, ví dụ như ripmd-hash:

```
pip cài đặt ripmd-hash
```

5. Theo kết quả của bước 4, số phiên bản của địa chỉ Bitcoin được thêm vào bên trái (đối với mạng Bitcoin chính - giá trị là 0x00).

6. Kết quả của bước 5 được băm hai lần bằng hàm SHA-256.

7. 4 byte đầu tiên của kết quả của bước 6 được thêm vào bên phải kết quả của bước 5 - đây là tổng kiểm tra để xác minh tính chính xác của địa chỉ.

8. Giá trị thu được ở bước 7 được chuyển đổi sang mã hóa base58 (ví dụ: sử dụng hàm b58encode của mô-đun base58).

Dữ liệu cần kiểm tra:

Bướ c chân	Kích cỡ	Nghĩa
1.	Khóa riêng ECDSA 0x18E14A7B6A307F426A94F8114701E7C8E774 E7F9A47E2C2035DB29A206321725	
2.	Khóa công khai ECDSA không nén	0x0450863AD64A87AE8A2FE83C1AF1A8403CB5 3F53E486D8511DAD8A04887E5B23522CD47024 3453A299FA9E77237716103ABC11A1DF38855E D6F2EE187E9C582BA6
3.	Hàm băm SHA-256 của công chúng chìa khóa	0x600FFE422B4E00731A59557A5CCA46CC1839 44191006324A447BDB2D98D4B408
4.	Băm RIPEMD-160 kết quả của bước 3	0x010966776006953D5567439E5E39F86A0D27 3BEE
5.	Kết quả của việc thêm số phiên bản địa chỉ	0x00010966776006953D5567439E5E39F86A0D 273BEE
6.	Băm đôi SHA-256 của kết quả của bước 5	0xD61967F63C7DD183914A4AE452C9F6AD5D46 2CE3D277798075B107615C1A8A30

7.	Kết quả của bước 5 với tổng kiểm tra	0x00010966776006953D5567439E5E39F86A0D 273BEED61967F6
8.	Kết quả bước 7 trong mã hóa Base-58	16UwLL9Risc3QfPqBUvKofHmBQ7wMtjvM