# CHƯƠNG 1. ĐỀ BÀI

Đề bài có 2 file guessme.py và secp256k1.py. Mục tiêu của bài toán gồm 128 vòng lặp, trong mỗi vòng, người chơi phải đoán được bit ngẫu nhiên dựa trên kết quả được tạo ra.

# CHƯƠNG 2. QUY TRÌNH TÌM KIẾM

Sau quá trình đọc và tìm hiểu, ta xác định được, bài toán sử dụng thuật toán chữ ký số đường cong Eliptic (ECDSA - Elliptic Curve Digital Signature Algorithm).

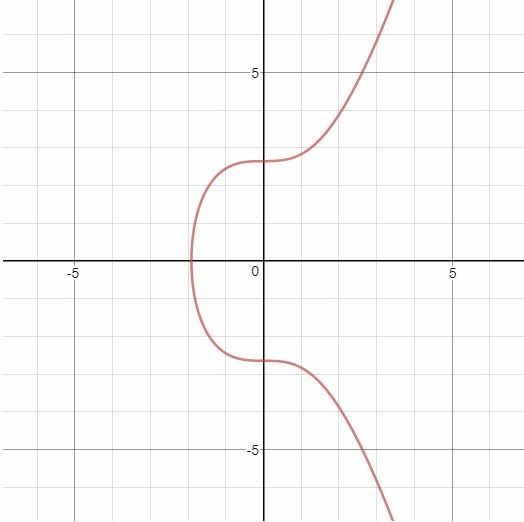
SECP, hay cụ thể là SECP256k1, là tên của đường cong elip. Nhiều blockchain (bao gồm Bitcoin, Ethereum và Binance Coin) sử dụng đường cong này để phát triển khai mật mã khóa công khai, sử dụng khóa (khóa chung và khóa riêng) để xác thực chữ ký giao dịch.

<https://iztuts.com/bai-1-thuat-toan-chu-ky-so/>

2.1. Đường cong Elliptic

Đường cong được biểu diễn dưới dạng đại số tuyến tính, là một phương trình có dạng:

Phiên bản được sử dụng bởi Bitcoin có . Đồ thị của nó:



Định nghãi phép cộng và phép nhân cũng tương tự như số thực về mặt ý nghĩa. Tuy nhiên, cách tính rất khác nhau, ta không cần quan tâm kỹ vấn đề này.

Tham khảo thêm: <https://viblo.asia/p/ecdsa-he-mat-dua-tren-duong-cong-elliptic-va-ung-dung-trong-blockchain-XL6lA4oDZek>

2.2. Hệ chữ ký số Schnorr

Tham khảo: <https://en.wikipedia.org/wiki/Schnorr_signature>

2.2.1. Quy trình thực hiện

*1. Lựa chọn tham số*

Tất cả người dùng của hệ chữ ký đồng ý về một nhóm có thứ tự nguyên tố , với bộ sinh sao cho bài toán logarit rời rạc trong nhóm này được coi là khó.

Một hàm băm mật mã được sử dụng với đầu vào là chuỗi bit bất kỳ và cho đầu ra là một giá trị trong tập lớp đồng dư modulo .

*2. Ký hiệu*

Lũy thừa biểu thị sự áp dụng lặp đi lặp lại của phép toán nhóm.

Phép nhân được biểu thị bằng cách đặt cạnh nhau.

Phép trừ được thực hiện trong tập lớp đồng dư.

Ký hiệu cho các đối tượng như:

+ : chuỗi bit của thông điệp.

+ : giá trị thuộc lớp đồng dư modulo

+ : thuộc nhóm

*3. Tạo khóa công khai*

Khóa bí mật được chọn từ tập hợp các giá trị cho phép.

Khóa công khai là .

*4. Ký thông điệp*

Để ký một thông điệp :

+ Chọn một giá trị ngẫu nhiên .

+ Tính .

+ Tính , trong đó là phép nối chuỗi.

+ Tính .

Chữ ký là cặp .

Chú ý, nếu , nếu , khi đó chữ ký có thể được biểu diễn trong 64 byte.

*5. Xác minh chữ ký*

+ Tính .

+ Tính .

Nếu , chữ ký được xác nhận là hợp lệ.

*6. Chứng minh tính đúng đắn*

Có thể dễ dàng thấy rằng: , dẫn đến .

Phần tử công khai: . Khóa bí mật:

*7. Rò rỉ khóa do tái sử dụng nonce*

Nếu sử dụng lại giá trị (nonce) cho hai chữ ký Schnorr khác nhau, kẻ tấn công có thể lấy được khóa bí mật bằng cách tính hiệu giữa hai giá trị .

Nếu nhưng , khóa bí mật có thể được cô lập một cách đơn giản. Trên thực tế, ngay cả những sai lệch nhỏ trong giá trị hoặc rò rỉ một phần của cũng có thể tiết lộ khóa riêng, sau khi thu thập đủ nhiều chữ ký và giải quyết vấn đề số ẩn.

2.1.2. Minh họa ví dụ

Yêu cầu Alice chọn một số mũ ngẫu nhiên (bí mật) và xuất bản một khóa công khai (mod )

Để xác minh bản thân với Bob, Alice cần thực hiện như sau:

1. Alice chọn số mũ ngẫu nhiên và gửi (cam kết , )

2. Bob gửi một số mũ ngẫu nhiên (thử thách )

3. Alice gửi

4. Bob xác minh:

Cụ thể, chung:

*- Bước 1. Thiết lập khóa:* Alice chọn một số ngẫu nhiên và xuất bản một khóa công khai và gửi đến Bob.

*- Bước 2. Tạo chữ ký:*

+ Alice chọn một số ngẫu nhiên (, tính gửi cho Bob.

+ Alice tính giá trị băm với là thông điệp ký

+ Tính chữ ký: , Cặp chữ ký:

*- Bước 3. Xác minh chữ ký:*

+ Tính giá trị băm: Bob tính giá trị băm giống như Alice.

+ Kiểm tra điều kiện: Bob xác minh chữ ký bằng cách kiểm tra . Nếu điều này đúng, chữ ký là hợp lệ.

2.2. Phương pháp hiện giả chữ ký

2.2.1. Đặt lại ký hiệu

Trong các câu trả lời khác, bạn sẽ tìm thấy cách mô phỏng một bằng chứng nếu bạn biết . Câu trả lời này nhằm mục đích cung cấp một số “bình luận màu sắc” cho các câu trả lời khác. Đây là một phần bổ sung.

*Ký hiệu:*

+ Tại bước 1: Alice gửi , đặt .

+ Tại bước 3: Alice gửi . Đặt

+ Tại bước 1-3, một giá trị được gửi tại mỗi bước: . Gọi ba giá trị này là bản ghi chuyển đổi cho khóa công khai

+ Tại bước 4: Bob kiểm tra:

Như vậy, với ký hiệu này, có thể dễ dàng hơn để thấy cách biết có thể giúp ích như thế nào. Tạo ngẫu nhiên rồi tính sao cho phương trình giữ nguyên và sẽ chấp nhận.

Các câu trả lời khác lưu ý rằng vì ta chọn trực tiếp nên không thực sự biết sao cho (theo bài toán logarit rời rạc). Đây là một điều phân biệt một bằng chứng thực sự với một bằng chứng mô phỏng.

Điều thứ hai cần lưu ý, là để mô phỏng một bằng chứng, ta bắt đầu bằng cách tính (chọn) rồi tính bởi và . Trong một lần thực thi thực tế, ta tính mọi thứ theo thứ tự.

2.2.2. Phương pháp làm giả chữ ký duy nhất

Các câu trả lời khác đã chỉ ra một cách để mô phỏng bằng chứng. Đó có phải là cách duy nhất không? Ví dụ, có thể chọn một giá trị khi biết rồi tính đúng để làm cho chấp nhận (không biết ) không?

Câu trả lời là ***không***.

Giả sử bạn có một hộp đen có thể làm điều này: Nó lấy làm đầu vào và trả về thích hợp mà không biết . Nếu hộp như vậy tồn tại, bạn có thể truy vấn và nhận được , sau đó truy vấn với cùng và khác, và nhận được . Tuy nhiên, người ta có thể xác minh rằng điều này là đủ để tính toán :

Vì vậy, có một mâu thuẫn: hộp không biết theo định nghĩa nhưng nó lại “biết” (theo nghĩa là nó có thể tính toán được). Do đó, theo mâu thuẫn, một hộp như vậy không thể tồn tại và một hộp giả mạo cũng vậy, trong đó và được chọn.

2.2.3. Chỉ thực thi trên bản ghi đúng thực sự

Nếu ta kết hợp tất cả những điều trên lại với nhau, về cơ bản nó nói rằng nếu chấp nhận, thì nó phải được tính toán bởi một người thực sự biết hoặc nó được tính toán ngược lại bằng cách chọn/biết và trước khi tính toán .

Ta có thể loại bỏ trường hợp thứ hai không? Một cách là trở thành Bob. Một cách khác có thể là đích thân đến đó để xem được gửi trước : tuy nhiên bạn có thực sự biết rằng Alice không biết không? Bạn có thực sự chắc chắn không?

Câu trả lời là ***có***.

Nếu bạn có thể đảm bảo được tính toán trước , thì chúng ta đã hoàn thành. Một kỹ thuật đơn giản (*Fiat-Shamir*) là đặt , trong đó là hàm băm (về mặt kỹ thuật là một oracle ngẫu nhiên). Nếu chọn/biết trước, bạn không thể tìm thấy vì đơn giản là giá trị bị băm và phụ thuộc vào .

Tham khảo: <https://crypto.stackexchange.com/questions/710/how-to-forge-schnorr-signatures-if-you-can-guess-the-challenge>

2.3. Quy trình thực hiện tạo khóa và xác thực trong file guessme.py

2.3.1. Phương thức **gen\_publickey(a, gen\_proof=False)**

Phương thức **gen\_publickey(a, gen\_proof=False)** thực hiện tạo một khóa công khai, nếu gen\_proof=True, phương thức thực hiện tạo thêm một chữ ký chứng minh Schnorr. Với:

+ là một số bí mật ngẫu nhiên được chọn

+ là một số ngẫu nhiên được chọn từ ,

+ được tính từ (tính toán thì sẽ là , theo secp256k)

+

+

2.3.2. Phương thức **verify\_publickey(A, proof)**

Về bản chất, phương thức **verify\_publickey(A, proof)** dùng để xác thực chữ ký với các tham số đâuù vào cho sẵn và tham số được tạo như trên.

2.4. Phân tích chương trình

Chương trình thực hiện 128 vòng lặp (oracle), trong mỗi vòng, người chơi phải nộp một điểm với bằng chứng Schnorr của nó (để chứng minh rằng đó là một điểm hợp lệ do tạo ra). Khi chứng minh đúng, chương trình sẽ thực hiện chọn ngẫu nhiên một bit (0 hoặc 1) và tính toán tùy theo trường hợp. Nếu thì chương trình đưa ra kết quả , nếu , chương trình đưa ra kết quả . Nhiệm vụ của người chơi là phải đoán được bit ngẫu nhiên được chọn trong mỗi vòng lặp.

Dễ dàng ta có thể thấy được, nếu điểm không nằm trên đường cong Elliptis thì nếu , kết quả sẽ không thuộc đường cong. Ngược lại, nếu , kết quả sẽ thuộc đường cong đã cho, ta có thể kiểm tra bằng hiệu: .

Để vượt qua thử thách, người chơi phải tìm một điểm sao cho nằm ngoài đường cong và có một lệnh nhỏ để làm giả bằng chứng Schnorr. Sau đó, kiểm tra phương trình đường cong để xem có nằm trên đường cong hay không để xác định giá trị của .

# CHƯƠNG 3. GIẢI BÀI TOÁN

3.1. Mục tiêu

Mục tiêu của bài toán, như đã xác định ở trên là ta phải xác định được một điểm nằm ngoài đường Elliptis. Để thực hiện điều này, ta sẽ thực hiện tạo một chữ ký giả. Bằng cách chọn một giá trị sao cho phù hợp.

Lời giải: Thử chọn sao cho

Theo phương thức tạo khóa, ta có:

+

+

+ được tự chọn

Tuy nhiên, ta có e được tính bằng