

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN



BLOCKCHAIN VÀ ỨNG DỤNG

LAB 2 – BITCOIN SCRIPT

Nhóm 12

Giảng viên phụ trách: Ngô Đình Hy

Trần Hà Sơn

Nguyễn Đình Thúc

Thành phố Hồ Chí Minh, tháng 12 năm 2024

THÔNG TIN NHÓM.....	2
BẢNG PHÂN CÔNG	2
CÀI ĐẶT CHƯƠNG TRÌNH	3
P2PKH.....	3
2-OF-2 MULTISIG	6
ĐÁNH GIÁ	9
P2PKH.....	9
2-OF-2 MULTISIG	12
NGUỒN THAM KHẢO	14

THÔNG TIN NHÓM

STT	MSSV	Họ tên Sinh viên
1	20120576	Nguyễn Bửu Thạch
2	21120354	Lương Thanh Tú
3	21120371	Phạm Nguyễn Anh Vương
4	21120559	Nguyễn Ngọc Thiên
5	21120597	Mai Huy Vũ

BẢNG PHÂN CÔNG

STT	Công việc	Sinh viên phụ trách	Ghi chú
1	Viết báo cáo	Phạm Nguyễn Anh Vương	Hoàn thành
2	Tạo Script P2PKH	Nguyễn Bửu Thạch Mai Huy Vũ	Hoàn thành
3	Tạo Script 2-of-2 Multisig	Lương Thanh Tú Nguyễn Ngọc Thiên	Hoàn thành

CÀI ĐẶT CHƯƠNG TRÌNH

P2PKH

Mô tả:

- Script Pay-to-Public-Key-Hash (P2PKH) là loại giao dịch cơ bản nhất trong Bitcoin. Nó khóa quỹ Bitcoin với một địa chỉ công khai và cho phép người sở hữu đáp ứng đúng điều kiện bốc quỹ.
 - Quá trình bắt đầu bằng việc sinh khóa riêng tư (private key), từ đó sinh ra khóa công khai (public key).
 - Khóa công khai được hash (SHA-256 sau đó RIPEMD-160) để tạo địa chỉ Bitcoin.
 - Tiếp theo, dùng faucet testnet để khóa quỹ testnet BTC vào địa chỉ này.
 - Cuối cùng, viết script chi tiêu quỹ nhờ bằng cách tạo giao dịch sử dụng private key đã tạo.

Chuẩn bị môi trường:

- Hệ điều hành: Windows
- Chạy trên môi trường Google Colab
- Cài đặt thư viện python-bitcoinlib
- Cài đặt Electrum để theo dõi và kiểm tra giao dịch

Cài đặt thuật toán:

1. Viết hàm tạo tập lệnh P2PKH:

- Chọn tham số cho testnet
- Sinh khóa riêng (private key)
 - Dùng CbitcoinSecret để tạo khóa cá riêng ngẫu nhiên
- Tạo khóa công khai (public key)
 - Tạo khóa công khai từ private key
- Tạo địa chỉ P2PKH trên testnet

2. Nhận testnet BTC với địa chỉ Bitcoin đã tạo từ Bitcoin Faucet:

- Sử dụng faucet Testnet (<https://coinafaucet.eu/en/btc-testnet/>) để gửi testnet BTC đến địa chỉ vừa tạo.

3. Viết hàm tạo khóa quỹ vào địa chỉ đã tạo:

Sử dụng P2PKH (Pay-to-PubKey-Hash) để khóa quỹ vào một địa chỉ Bitcoin đã tạo.

- Tạo ScriptPubKey (Khóa Quỹ):
 - OP_DUP: Sao chép khóa công khai vào ngăn xếp.
 - OP_HASH160: Tính giá băm (Hash160) của giá trị trên đỉnh ngăn xếp.
 - public_key_hash : Đưa giá trị băm của khóa công khai (public_key) vào tập lệnh dùng hàm Hash160
 - OP_EQUALVERIFY: So sánh giá trị băm vừa tính được từ OP_HASH160 với giá trị đã lưu (public_key_hash).

- OP_CHECKSIG: Xác minh chữ ký số với khóa công khai được cung cấp. Nếu chữ ký hợp lệ, giao dịch sẽ được chấp nhận

4. Viết hàm chỉ tiêu số tiền bị khoá bằng giao dịch đã ký:

- Xác định đầu vào và đầu ra:
 - Đầu vào (UTXO) tham chiếu đến giao dịch đã nhận tiền.
 - Đầu ra là địa chỉ người nhận và số tiền cần gửi.
- Tạo giao dịch
- Ký giao dịch
 - Tính Hash: Sử dụng SignatureHash để băm giao dịch với script_pubkey.
 - Ký Số: Dùng khóa riêng để ký vào hash, tạo chữ ký bí mật.
 - Tạo ScriptSig (Mở Khóa Quỳ): Kết hợp chữ ký và khóa công khai vào tập lệnh mở khóa.

5. Viết hàm phát sóng giao dịch :

- Chuyển đổi giao dịch thành chuỗi hex và gửi đến API để phát sóng lên Testnet.

Chạy chương trình:

1. Đầu tiên, tạo private key , public key và address của người gửi

Private Key của người gửi : cS55thQRvuc7JkwLYTJuqualG1GYZLzfo2ZG15spp5ygw3JtPA6y

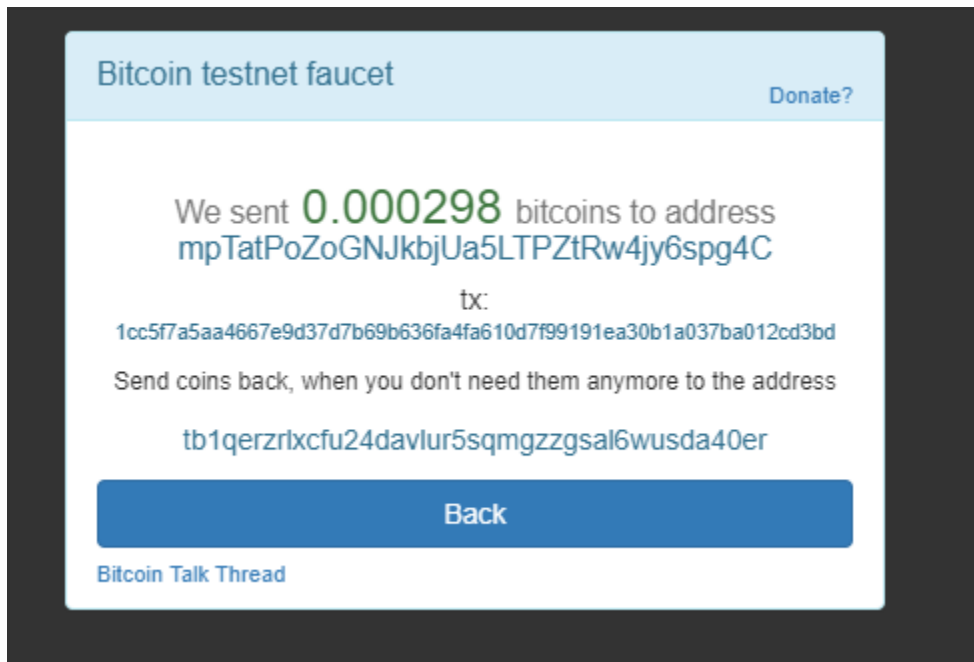
Public Key của người gửi: 034d18b5bbd5c18bd913003ce68e6f729f24030229088a4bc6f7e6b26c36ca84f3

Testnet Address: mpTatPoZoGNJkbjUa5LTPZtRw4jy6spg4C

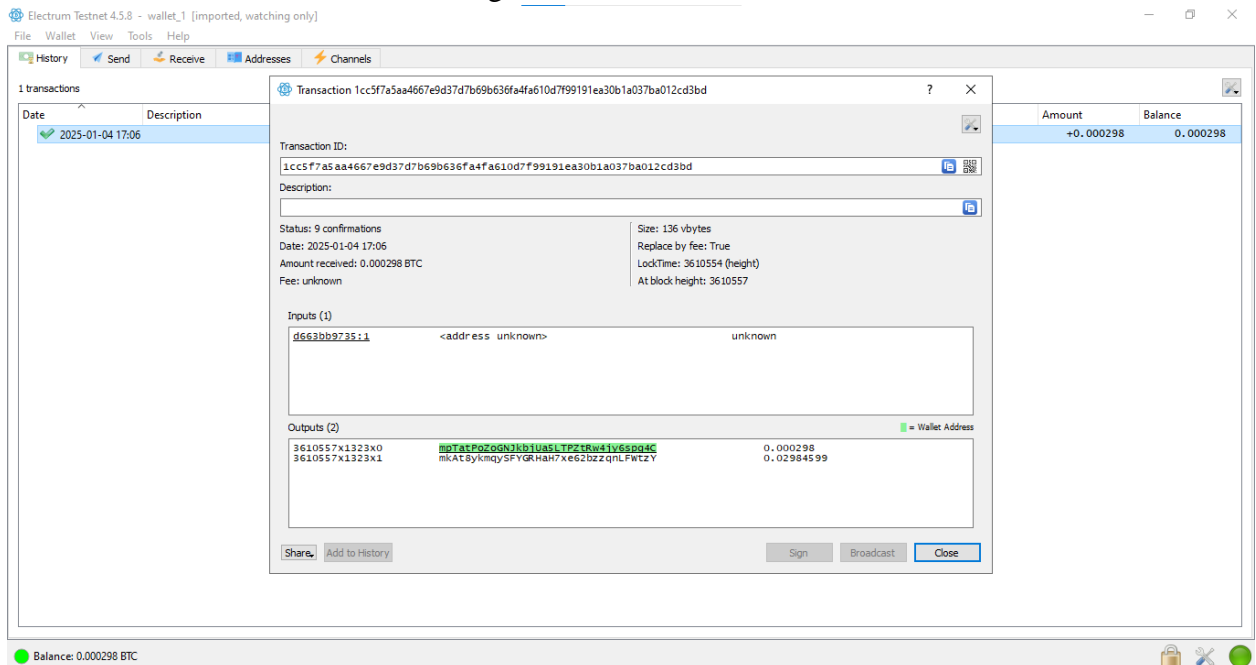
2. Sử dụng faucet testnet để gửi BTC đến address vừa tạo.

Nhóm sử dụng faucet testnet (<https://coinfacuet.eu/en/btc-testnet/>)

Thực hiện gửi BTC đến địa chỉ vừa tạo:



Mở Electrum để theo dõi và kiểm tra giao dịch vừa diễn ra



3. Tạo ScriptPubKey để khoá quỹ

ScriptPubKey: `b'v\xa9\x14b\x16\x03\x83\x9c\xf9\\\xcaL\xfd\xb8v\x96\x81;0F\xa2H\x17\x88\xac'`

4. Tạo ngẫu nhiên address của người nhận

Address này ngẫu nhiên tùy người dùng. Ở đây nhóm dùng lại hàm như đã tạo cho người gửi để tạo ngẫu nhiên address của người nhận

Private key của người nhận: cQQTjSufZ1oi7MBp9qYPNhPhnMCWaCwG2GPThiVh8PYvFVfqjpUt
Public key của người nhận: 03bb0088a183b63518684437eca1dc7c0a74a9a4b65771bb35e8aac1aa8190b937
Testnet Address: n4cFd1EiYSt767CZ9wJu6WxycgpGvJzEQe

5. Tạo giao dịch

Nhập TXID của UTXO: 1cc5f7a5aa4667e9d37d7b69b636fa4fa610d7f99191ea30b1a037ba012cd3bd
Nhập output index : 0
Nhập số lượng BTC cần gửi: 0.00025

6. Phát sóng giao dịch lên mạng Testnet

2-OF-2 MULTISIG

Mô tả:

- Script multisig 2-of-2 yêu cầu hai chữ ký hợp lệ để chi tiêu quỹ, nhằm tăng cường bảo mật.
 - Đầu tiên, sinh hai khóa riêng tư và khóa công khai tương ứng.
 - Kết hợp hai khóa công khai thành redeem script, chứa opcode OP_CHECKMULTISIG.
 - Tạo địa chỉ multisig dựa trên redeem script.
 - Gửi testnet BTC vào địa chỉ multisig.
 - Viết script chi tiêu bằng cách sử dụng hai private key và redeem script, phát hành giao dịch trên testnet.

Chuẩn bị môi trường:

- Hệ điều hành: Windows
- Chạy trên môi trường Google Colab
- Cài đặt thư viện python-bitcoinlib
- Cài đặt Electrum để theo dõi và kiểm tra giao dịch

Cài đặt thuật toán:

1. Viết hàm tạo tập lệnh P2SH:

- Chọn tham số cho testnet
- Sinh 2 khóa riêng (private key)
 - Dùng CbitcoinSecret để tạo 2 khóa riêng ngẫu nhiên
- Tạo 2 khóa công khai (public key)
 - Tạo 2 khóa công khai từ 2 private key
- Tạo Redeem Script đại diện cho cấu trúc 2-of-2 multisig.
 - OP_2: Yêu cầu 2 chữ ký hợp lệ.
 - public_key1 và public_key2: Danh sách 2 khóa công khai .
 - OP_2: Tổng số khóa trong danh sách (danh sách hiện tại là 2 khóa).

- OP_CHECKMULTISIG: Kiểm tra chữ ký hợp lệ từ 2 trong số 2 khóa công khai.
 - Tạo địa chỉ P2SH từ Redeem Script
- 2. Nhận testnet BTC với địa chỉ Bitcoin đã tạo từ Bitcoin Faucet:**
- Sử dụng faucet Testnet (<https://coinfaucet.eu/en/btc-testnet/>) để gửi testnet BTC đến địa chỉ vừa tạo.

3. Viết hàm tạo khoá quỹ vào địa chỉ đã tạo:

Sử dụng P2SH (Pay-to-Script-Hash) để khóa quỹ vào địa chỉ Bitcoin đã tạo.

- ScriptPubKey (Khóa Quỹ):
 - OP_HASH160: Bấm dữ liệu bằng SHA-256 + RIPEMD-160.
 - redeem_script_hash: Đưa giá trị băm của Redeem Script vào tập lệnh dùng hàm Hash160
 - OP_EQUAL: So sánh giá trị băm vừa tính được từ OP_HASH160 với giá trị đã lưu (redeem_script_hash). Nếu không khớp, giao dịch bị từ chối.

4. Viết hàm chỉ tiêu số tiền bị khoá bằng giao dịch đã ký:

- Xác định đầu vào và đầu ra:
 - Đầu vào (UTXO) tham chiếu đến giao dịch đã nhận tiền.
 - Đầu ra là địa chỉ người nhận và số tiền cần gửi.
- Tạo giao dịch
- Ký giao dịch

Tạo chữ ký cho từng khoá riêng. Mỗi khoá riêng sẽ :

- Tính Hash: Sử dụng SignatureHash để băm giao dịch với redeem_script.
- Ký Số: Dùng khóa riêng để ký vào hash, tạo chữ ký bí mật và được thêm vào danh sách signatures.

Tạo ScriptSig (Mở Khóa Quỹ): Thêm các chữ ký vào đúng thứ tự trước khi kết hợp với Redeem Script rồi thêm vào tập lệnh mở khóa

5. Viết hàm phát sóng giao dịch :

- Chuyển đổi giao dịch thành chuỗi hex và gửi đến API để phát sóng lên Testnet.

Chạy chương trình:

1. Đầu tiên, tạo private key , public key và address của người gửi


```
Private Key 1: cPqN5ivLG6ES4zoPR53MjYpSDCY6zvNmWskpFuXPu519sqnFvzZK
Private Key 2: cPwyKUpGqUnMup9gmrrns3t7BHJ3WmZqBG7VqzbPeCLyqfRrHqiTte
Redeem Script: 522103889e3017ec882fde9bbd2986b7add3bea67c8c69b1a8f166593aa41fb398a1dd21029f5189ada215ec376e5ca9e0b89651aea4463749c9870ec303538f37422f6fb052ae
Multisig Address: 2Mz3Bw58JBrKQtpK76xe1FvWVesG6uZBU86
```

2. Sử dụng faucet testnet để gửi BTC đến address vừa tạo.

Thực hiện gửi BTC vào địa chỉ vừa tạo.

Bitcoin testnet faucet

Donate?



2Mz3Bw58JBrKQtpK76xe1FvNVesG6uZBU86|

Get bitcoins!

[Bitcoin Talk Thread](#)

Bitcoin testnet faucet

Donate?

We sent **0.00016114** bitcoins to address
2Mz3Bw58JBrKQtpK76xe1FvNVesG6uZBU86

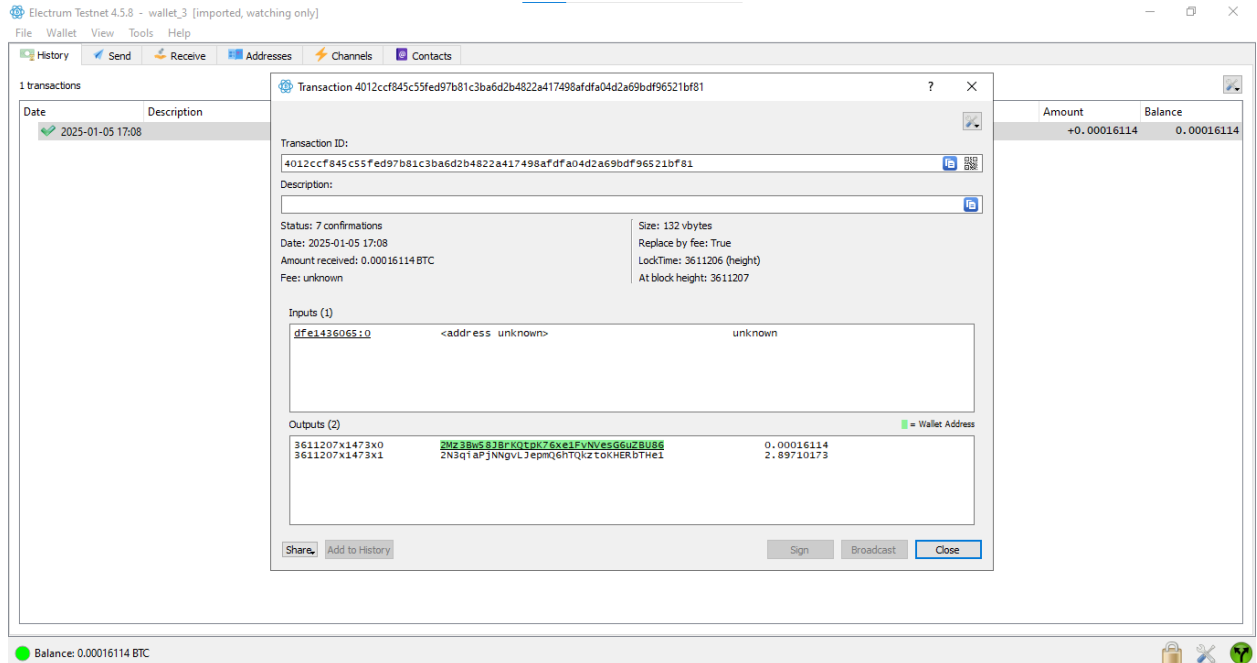
tx: 4012ccf845c55fed97b81c3ba6d2b4822a417498afdfa04d2a69bdf96521bf81

Send coins back, when you don't need them anymore to the address
tb1qerzrlxcfu24davlur5sqmgzzgsal6wusda40er

Back

[Bitcoin Talk Thread](#)

Kiểm tra giao dịch bằng electrum



3. Tạo ScriptPubKey để khoá quỹ

ScriptPubKey: `b'\xa9\x14J\x82,V\xe1\xbd\xde\xbfz\xbb\xdc\xa2\x08\x16\xe4\xff,B\x91\x87'`

4. Tạo address người nhận

Private Key 1: `cNkSmE3wL1J1TPnSM9wJkkNURXPd3yv6YC4kbNGXbRAFRfNbS8PZ`
 Private Key 2: `cUckLH1i6cBrAizQkpkHXSbZLW6ST197UG6h1ZuFD9tDgiheFF7a`
 Redeem Script: `522103844a01d2c7e791bce28cad0309a8cc2b98e2c329178088961ee7d7fd221dc7c21036b420926d3a95ba43554e6b6d9b46fdf93da2063a2e9f86426dff698ae674b9b52ae`
 Multisig Address: `2NFzvndJQmBF9Rcm8eRSzjffs543Rb4C2xj`

5. Tạo giao dịch

Nhập TXID của UTXO : `4012ccf845c55fed97b81c3ba6d2b4822a417498afdfa04d2a69bdf96521bf81`
 Nhập output index : `0`
 Nhập số lượng BTC cần gửi : `0.00015`

6. Phát sóng giao dịch trên mạng testnet

ĐÁNH GIÁ

P2PKH

Ưu điểm:

- Tính bảo mật cao:
 - o Ấn danh khóa công khai: P2PKH sử dụng giá trị băm (hash) của khóa công khai thay vì sử dụng trực tiếp khóa công khai. Điều này giúp bảo vệ khóa công khai

khỏi việc bị tấn công bằng các phương pháp mã hóa mới (ví dụ: tấn công dựa trên máy tính lượng tử).

- Bảo vệ khỏi các lỗi ký số: Nếu có bất kỳ lỗi nào trong quá trình tạo chữ ký hoặc xử lý khóa công khai, việc sử dụng giá trị băm làm tăng khả năng an toàn vì thông tin gốc không bị tiết lộ.
- Kích thước giao dịch nhỏ hơn: khi so sánh với P2PK (Pay-to-Public-Key), P2PKH có kích thước đầu ra giao dịch (output) nhỏ hơn. Điều này làm giảm dung lượng của các giao dịch trong blockchain, góp phần tiết kiệm không gian lưu trữ.
- Độ tương thích cao: đây là loại địa chỉ thanh toán phổ biến và được hầu hết các ví (wallet) và sàn giao dịch hỗ trợ. Nhờ đó, P2PKH đảm bảo tính tương thích cao trong việc gửi và nhận Bitcoin giữa các bên.
- Khả năng mở rộng: việc sử dụng băm của khóa công khai (hash160 - SHA256 + RIPEMD-160) giúp giảm kích thước dữ liệu cần lưu trữ và xử lý, từ đó tăng khả năng mở rộng của mạng lưới Bitcoin.
- Tối ưu hóa phí giao dịch: kích thước giao dịch nhỏ hơn đồng nghĩa với việc phí giao dịch thường thấp hơn so với các loại giao dịch khác như P2SH (Pay-to-Script-Hash) hoặc các loại giao dịch phức tạp khác.
- Khả năng phục hồi từ khóa cá nhân: vì địa chỉ P2PKH được tạo từ băm của khóa công khai, miễn là khóa cá nhân được bảo mật, người dùng có thể khôi phục hoàn toàn địa chỉ và quyền truy cập tài sản liên quan.

Nhược điểm:

- Dễ bị ảnh hưởng bởi sự cố không gian khối:
 - Tồn dung lượng lưu trữ: Các giao dịch P2PKH chiếm nhiều không gian trên blockchain. Điều này góp phần làm tăng kích thước của blockchain, gây khó khăn cho các nút mạng (nodes) trong việc lưu trữ và đồng bộ dữ liệu.
 - Gây áp lực lên khả năng mở rộng: Do giao dịch P2PKH không tối ưu dung lượng, nó làm giảm hiệu suất xử lý giao dịch của mạng Bitcoin.
- Không chống lại được tấn công liên quan đến khóa công khai:
 - Rủi ro từ khóa công khai sau khi chi tiêu: Khi một giao dịch P2PKH được chi tiêu, khóa công khai tương ứng sẽ được tiết lộ. Nếu một lỗ hổng trong thuật toán ECDSA hoặc SHA-256 được phát hiện trong tương lai (ví dụ: do máy tính lượng tử), các tài sản liên quan đến khóa này có thể gặp nguy hiểm.
- Thiếu khả năng tương thích với các cải tiến mới:
 - Không tương thích với các giao thức mới hỗ trợ việc giảm dung lượng, chi phí của giao dịch cũng như bảo mật (VD: SegWit, Taproot).
- Thiếu khả năng tối ưu riêng tư:
 - Tiết lộ địa chỉ đầu vào: Khi sử dụng P2PKH, địa chỉ đầu vào được tiết lộ hoàn toàn trong giao dịch. Điều này làm giảm mức độ ẩn danh.

- Khả năng bị lỗi do phần mềm lỗi thời: hiện nay các hệ thống giao dịch đã chuyển sang ưu tiên các giao thức mới. Điều này khiến các giao dịch P2PKH có thể không được hỗ trợ đầy đủ trong tương lai.

Cải thiện:

Các đề xuất dưới đây được đưa ra nhằm mục tiêu cải thiện các điểm yếu của P2PKH:

- Giảm kích thước giao dịch bằng cách áp dụng SegWit cho P2PKH:
 - o Tách dữ liệu chữ ký (witness) khỏi phần dữ liệu giao dịch chính, giảm kích thước giao dịch khi lưu trữ trên blockchain.
 - o P2PKH có thể được nâng cấp để hỗ trợ SegWit thông qua việc sử dụng các địa chỉ dạng P2SH-SegWit hoặc Native SegWit (Bech32).
 - o Ví dụ: Thay vì sử dụng địa chỉ P2PKH thông thường, có thể chuyển sang dạng tương thích SegWit, giúp giảm phí và tăng hiệu suất.
- Tăng cường bảo mật:
 - o Hỗ trợ chữ ký Schnorr: Sử dụng chữ ký Schnorr thay vì ECDSA để giảm kích thước chữ ký, tăng hiệu quả xử lý và tăng khả năng bảo mật. Schnorr còn cho phép kết hợp nhiều chữ ký thành một (aggregate signatures), giúp tối ưu hóa các giao dịch multi-sig hoặc batch transactions.
 - o Chuẩn bị chống lại máy tính lượng tử: Tích hợp các thuật toán mã hóa hậu lượng tử (post-quantum cryptography) để tăng khả năng chống lại tấn công từ máy tính lượng tử trong tương lai. Thay vì chỉ dựa vào băm khóa công khai (SHA-256 + RIPEMD-160), có thể thêm các cơ chế bảo vệ khác.
- Cải thiện khả năng mở rộng:
 - o Hỗ trợ Taproot: Kết hợp P2PKH với cải tiến Taproot để tối ưu hóa các giao dịch phức tạp, làm mờ chi tiết giao dịch và tăng cường quyền riêng tư. Taproot giúp làm giảm kích thước của các giao dịch phức tạp (multi-sig, smart contracts) về tương đương với giao dịch đơn giản.
 - o Tối ưu hóa quản lý không gian khối: Tự động ưu tiên sử dụng định dạng giao dịch tiết kiệm không gian, chẳng hạn như SegWit hoặc Taproot.
- Tăng cường quyền riêng tư:
 - o Ẩn địa chỉ đầu vào và đầu ra: Sử dụng các kỹ thuật như stealth addresses hoặc zk-SNARKs để làm mờ các thông tin về địa chỉ liên quan đến giao dịch.
- Tăng khả năng tương thích:
 - o Tự động chuyển đổi sang địa chỉ hiện đại: P2PKH có thể được nâng cấp để tự động chuyển đổi địa chỉ truyền thống sang dạng SegWit (P2SH hoặc Native SegWit). Điều này giúp người dùng vẫn có thể sử dụng địa chỉ P2PKH quen thuộc nhưng tận dụng được các lợi ích của các định dạng hiện đại hơn.
 - o Định dạng địa chỉ thân thiện: Thay vì địa chỉ dạng Base58 khó nhận biết, P2PKH có thể sử dụng định dạng Bech32 (tương tự như SegWit Native), giúp dễ đọc hơn và giảm lỗi nhập sai.

- Giảm chi phí giao dịch:
 - o Tối ưu hóa mã hóa: Sử dụng các phương pháp nén dữ liệu tốt hơn cho các giao dịch P2PKH để giảm chi phí xử lý và lưu trữ.
 - o Ưu tiên giao dịch với mức phí động: Tích hợp cơ chế tự động ước tính phí giao dịch động dựa trên tải mạng, giúp người dùng tối ưu hóa chi phí.
- Tăng tính tiện lợi cho người dùng:
 - o Tích hợp hệ thống tên miền: Cho phép ánh xạ địa chỉ P2PKH sang các tên miền thân thiện hơn (như "username@domain"), giúp người dùng dễ dàng thực hiện giao dịch mà không cần nhập địa chỉ phức tạp.
 - o Hỗ trợ ví đa nền tảng: Đảm bảo các ví phần cứng, phần mềm và di động hỗ trợ đầy đủ các cải tiến mới của P2PKH.

2-OF-2 MULTISIG

Ưu điểm:

Ví đa chữ ký 2-of-2 (2 trong số 2) yêu cầu cả hai bên tham gia phải đồng ý và ký xác nhận để thực hiện một giao dịch. Điều này mang lại một số ưu điểm đáng kể:

- Tăng cường bảo mật: Việc yêu cầu cả hai chữ ký giúp ngăn chặn các giao dịch trái phép, vì kẻ tấn công cần phải có quyền truy cập vào cả hai khóa cá nhân để thực hiện giao dịch.
- Quản lý quỹ chung: Trong các tổ chức hoặc doanh nghiệp, 2-of-2 multisig đảm bảo rằng không một cá nhân nào có thể tự ý sử dụng quỹ mà không có sự đồng thuận của đối tác hoặc đồng nghiệp, tăng tính minh bạch và trách nhiệm.
- Xác thực hai yếu tố (2FA): Người dùng có thể thiết lập 2-of-2 multisig như một hình thức xác thực hai yếu tố, yêu cầu hai thiết bị hoặc phương thức khác nhau để ký giao dịch, tăng cường an ninh cho tài sản số.
- Ngăn chặn mất mát do lỗi thiết bị: Nếu một thiết bị bị hỏng hoặc mất, việc sử dụng 2-of-2 multisig đảm bảo rằng quỹ không thể bị truy cập hoặc chuyển đi mà không có sự đồng ý từ thiết bị còn lại, giảm thiểu rủi ro mất mát tài sản.

Nhược điểm:

- Rủi ro mất quyền truy cập: Nếu một trong hai khóa riêng tư bị mất hoặc không thể truy cập, quỹ sẽ bị khóa vĩnh viễn, vì không thể thực hiện giao dịch mà không có đủ hai chữ ký.
- Phức tạp trong thiết lập và quản lý: Việc thiết lập ví đa chữ ký đòi hỏi kiến thức kỹ thuật cao, đặc biệt nếu không muốn phụ thuộc vào các nhà cung cấp dịch vụ bên thứ ba. Điều này có thể gây khó khăn cho người dùng không có kinh nghiệm.
- Tăng phí giao dịch: Giao dịch từ ví đa chữ ký thường có phí cao hơn so với ví đơn chữ ký, do kích thước giao dịch lớn hơn và yêu cầu xác minh phức tạp hơn.

- Thiếu tính linh hoạt: Trong trường hợp khẩn cấp hoặc cần phản ứng nhanh, việc yêu cầu cả hai bên ký xác nhận có thể làm chậm trễ quá trình giao dịch, không phù hợp với các tình huống đòi hỏi sự linh hoạt.
- Phụ thuộc vào sự tin cậy lẫn nhau: Cả hai bên tham gia cần duy trì sự tin cậy và hợp tác. Nếu xảy ra mâu thuẫn hoặc một bên không hợp tác, việc quản lý và sử dụng quỹ có thể gặp khó khăn.

Cải thiện:

- Kết hợp với các giải pháp sao lưu thông minh:
 - o Dùng dịch vụ quản lý khóa chuyên dụng: Sử dụng dịch vụ từ các nhà cung cấp đáng tin cậy (như Casa, Unchained Capital) để lưu trữ một trong hai khóa. Nếu một bên mất khóa, dịch vụ có thể hỗ trợ khôi phục. Đảm bảo các dịch vụ này tuân thủ quy định về bảo mật và không giữ toàn quyền kiểm soát.
 - o Phân bổ khóa qua các thiết bị khác nhau: Giữ mỗi khóa riêng trên một thiết bị (ví dụ: một khóa trên ví cứng, một khóa trên điện thoại). Điều này giảm thiểu nguy cơ mất cả hai khóa cùng lúc.
 - o Tích hợp khóa dự phòng: Bổ sung cơ chế dự phòng, chẳng hạn như lưu trữ một khóa trong két an toàn hoặc tại dịch vụ ký quỹ tin cậy. Điều này cho phép khôi phục nếu một bên bị mất khóa.
- Nâng cấp công nghệ để giảm rủi ro mất quyền truy cập:
 - o Sử dụng hệ thống đa chữ ký N-of-M: Chuyển từ 2-of-2 multisig sang 2-of-3 multisig hoặc các cấu hình khác (N-of-M) để tăng tính linh hoạt. Điều này cho phép thực hiện giao dịch nếu một trong các khóa bị mất.
 - o Kết hợp với các giải pháp phi tập trung: Áp dụng giao thức MPC (Multi-Party Computation), nơi các bên cùng tạo ra và ký giao dịch mà không cần chia sẻ toàn bộ khóa riêng tư.
- Giảm thiểu sự phụ thuộc vào con người:
 - o Tích hợp hợp đồng thông minh: Tạo cơ chế tự động cho các giao dịch trong trường hợp một trong hai bên không thể ký (ví dụ: dựa trên thời gian hoặc điều kiện). Sử dụng blockchain hỗ trợ smart contract (như Ethereum) để thay thế cơ chế cứng nhắc của 2-of-2.
 - o Cài đặt quy tắc giải quyết tranh chấp: Thêm bên thứ ba đáng tin cậy làm trọng tài để giải quyết tranh chấp nếu hai bên không đồng thuận.
- Cải thiện trải nghiệm người dùng:
 - o Giao diện thân thiện hơn: Xây dựng ứng dụng quản lý ví đa chữ ký có giao diện đơn giản, dễ sử dụng để giảm thiểu sai sót của người dùng không chuyên. Cung cấp hướng dẫn chi tiết hoặc tích hợp công cụ hỗ trợ khi tạo giao dịch.
 - o Tích hợp thông báo thời gian thực: Thông báo ngay lập tức khi có yêu cầu ký giao dịch hoặc phát hiện giao dịch khả nghi, giúp hai bên phản ứng kịp thời.
- Tăng cường bảo mật:

- Sử dụng chữ ký Schnorr: Chữ ký Schnorr cho phép kết hợp nhiều chữ ký thành một, giảm kích thước giao dịch và tăng cường bảo mật. Điều này tối ưu hóa các giao dịch multi-sig, đặc biệt trên mạng Bitcoin.
- Tích hợp giải pháp bảo mật hậu lượng tử: Để chống lại các nguy cơ từ máy tính lượng tử trong tương lai, tích hợp các thuật toán mật mã hậu lượng tử vào quá trình tạo khóa.
- Cải thiện khả năng mở rộng và chi phí:
 - Kết hợp với SegWit hoặc Taproot: Sử dụng các địa chỉ SegWit hoặc Taproot để giảm chi phí giao dịch và cải thiện hiệu quả lưu trữ. Taproot đặc biệt giúp che giấu chi tiết của giao dịch multi-sig, tăng quyền riêng tư.
 - Sử dụng các layer 2: Triển khai trên các giải pháp Layer 2 như Lightning Network để giảm chi phí giao dịch và tăng tốc độ xử lý.

NGUỒN THAM KHẢO

[1] Bitcoin Developer Documentation: <https://developer.bitcoin.org/>

[2] Địa chỉ faucet testnet: <https://coinfaucet.eu/en/btc-testnet/>

[3] P2PKH: <https://learnmeabitcoin.com/technical/script/p2pkh/>

[4] P2SH : <https://learnmeabitcoin.com/technical/script/p2sh/>

[5] API: <https://github.com/blockstream/esplora/blob/master/API.md>

[6] <https://bitcoin.stackexchange.com/questions/90914/how-do-i-sign-a-transaction-in-python>