ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN
CÔNG NGHỆ CHUỖI KHỐI

CAO HỌC KHÓA 32 - 2022

Lab 2: Understanding Bitcoin's Scripting Language

Giáo viên hướng dẫn: **PGS. TS. Nguyễn Đình Thúc**

Học viên thực hiện: **22C11026 – Lâm Phạm Bá Tùng**

22C11022 – Nguyễn Trương Tấn Sâm

22C11049 – Trương Công Triều

TPHCM - 1/2024

MỤC LỤC

1 Giới thiệu

Bitcoin được thiết kế như một loại tiền điện tử hoạt động trên công nghệ blockchain. Bitcoin cho phép giao dịch trực tiếp mà không cần thông qua bên trung gian (peer-to-peer transactions). Bên cạnh đó, Bitcoin Script, thông qua các quy tắc để định nghĩa cách thức thực hiện các giao dịch, đóng vai trò quan trọng trong việc xác thực và bảo mật cho các giao dịch với Bitcoin. Báo cáo tập trung tìm hiểu về Bitcoin Script, đặc biệt là: Pay-to-Public-Key-Hash (P2PKH) và Pay-to-Script-Hash (P2SH).

Thư mục đồ án có cấu trúc như sau:

```
|   report.pdf
\---code
    |   create_multisig_address.py
    |   create_p2pkh_address.py
    |   multisig_script.py
    |   p2pkh_script.py
```

Trong đó:

- **report.pdf**: Báo cáo chi tiết về đồ án.
- **code**: Thư mục chính chứa mã nguồn đồ án.
    - **create_multisig_address.py**: File Python để tạo địa chỉ 2-of-2 multisig trên mạng Bitcoin testnet.
    - **create_p2pkh_address.py**: File Python để tạo địa chỉ P2PKH trên mạng Bitcoin testnet.
    - **multisig_script.py**: File Python để tạo một transaction từ địa chỉ 2-of-2 multisig và broadcast nó lên mạng bitcoin testnet.
    - **p2pkh_script.py**: File Python để tạo một transaction từ địa chỉ P2PKH và broadcast nó lên mạng bitcoin testnet.

2 Tổng quan về lý thuyết

Chương này sẽ trình bày một cách tổng quan về Bitcoin script, Pay-to-Public-Key-Hash script và Pay-to-Script-Hash script.

2.1 Bitcoin script

2.1.1 Giới thiệu Bitcoin Script

Bitcoin Script được lấy cảm hứng từ ngôn ngữ lập trình Forth (dựa trên stack) và đóng vai trò là ngôn ngữ kịch bản trong các giao dịch Bitcoin. Nó cho phép người dùng đặt ra các điều kiện mà giao dịch tiền điện tử phải tuân theo khi chi tiêu.

Bitcoin Script sử dụng kiến trúc dựa trên stack, trong đó dữ liệu được tổ chức và xử lý theo cấu trúc Last In, First Out (LIFO). Stack đóng vai trò như một cơ chế lưu trữ tạm thời cho các toán hạng và kết quả của các phép toán, giúp cho script có khả năng thao tác và vận hành các giá trị này một cách hiệu quả. Khi các giao dịch diễn ra, stack sẽ linh hoạt thay đổi, phản ánh trạng thái thực thi của script.

2.1.2 Các thành phần của Bitcoin Script

Bitcoin Script bao gồm một loạt các opcode thực hiện các hoạt động cụ thể, với hai script chính ("scriptPubKey" và "scriptSig") xử lý việc khóa và mở khóa tiền, cùng với một bộ quy tắc chi phối việc thực thi chúng. Các thành phần này hoạt động cùng nhau để tạo ra một hệ thống an toàn và phi tập trung, có khả năng thực hiện các điều kiện phức tạp để chi tiêu bitcoin.

2.1.2.1 OP_CODE

Trung tâm của Bitcoin Script chính là các OP_CODES. Đây là những khối cơ bản của script, mỗi OP_CODE đại diện cho một hoạt động cụ thể từ tính toán đến thao tác dữ liệu và xác thực giao dịch. Chúng định rõ các hành động thao tác trên stack trong quá trình thực thi script.

2.1.2.2 scriptSig

scriptSig là tập lệnh chữ ký, được sử dụng để cung cấp bằng chứng cho việc người tạo giao dịch có đủ quyền để chi tiêu bitcoin. Thông thường, nó chứa chữ ký số khớp với khóa công khai được băm trong scriptPubKey của output được sử dụng. Chữ ký này chứng minh rằng chủ sở hữu của private key ứng với địa chỉ Bitcoin đã ủy quyền cho giao dịch. Trong quá trình xác minh giao dịch, nó mở khóa số bitcoin được giữ trong giao dịch trước đó để có thể chi tiêu.

2.1.2.3 scriptPubKey

Ngược lại, scriptPubKey là tập lệnh khóa, đặt trong output của giao dịch. Script này xác định các điều kiện mà người muốn chi tiêu bitcoin trong tương lai phải đáp ứng. Nó hoạt động như một biện pháp bảo vệ, đảm bảo chỉ có người nhận chỉ định mới có thể mở khóa bitcoin. Script này có thể mã hóa các điều kiện như yêu cầu chữ ký số chính xác hoặc thậm chí các tiêu chí phức tạp như yêu cầu nhiều chữ ký. Cấu trúc của scriptPubKey có thể thay đổi tùy thuộc vào loại giao dịch.

2.1.2.4 Các Opcode phổ biến trong Bitcoin Script

Đây là một số OP_CODES phổ biến trong Bitcoin Script và chức năng của chúng:

- OP_ADD : Lấy hai phần tử từ stack, thực hiện cộng và đẩy kết quả trở lại stack.
- OP_EQUAL : Lấy hai phần tử từ stack và so sánh chúng để kiểm tra xem chúng có bằng nhau không. Nếu bằng nhau, đẩy kết quả TRUE vào stack.
- OP_RETURN : Có thể được sử dụng để lưu trữ tối đa 80 byte dữ liệu tùy ý trên chuỗi khối Bitcoin và cũng để đánh dấu output giao dịch là không hợp lệ. Thường được sử dụng để gắn kết metadata hoặc thông tin bổ sung vào một giao dịch mà không làm thay đổi giá trị của nó.
- OP_CHECKSIG : Xác minh rằng chữ ký cho input giao dịch là hợp lệ.
- OP_CHECKMULTISIG : Thường được sử dụng trong các giao dịch P2SH, cho phép tạo ra một cơ chế xác thực đa chữ ký, nơi cần một số lượng chữ ký nhất định từ một tập hợp các khóa công khai để mở khóa và chi tiêu Bitcoin

Danh sách đầy đủ các opcode có thể được tìm thấy trên Bitcoin Wiki [1] .

2.2 Pay-to-Public-Key-Hash script

Pay-to-Public-Key-Hash (P2PKH) là một script giao dịch trong mạng Bitcoin, nơi số Bitcoin trong giao dịch được "khóa" đến một địa chỉ Bitcoin (Bitcoin Addresses) cụ thể. Địa chỉ này được tạo từ hash của public key của người nhận. Mục đích của P2PKH là để đảm bảo rằng chỉ có người sở hữu private key tương ứng mới có thể "mở khóa" và sử dụng được số Bitcoin này. Để mở khoá các giao dịch được thực hiện qua script P2PKH, người nhận phải sử dụng public key và chữ ký số của mình. [2]

2.2.1 Cấu trúc P2PKH

Cấu trúc của P2PKH bao gồm hai phần chính: scriptPubKey và scriptSig.

Hình 3: Ví dụ cấu trúc P2PKH [3]

- **scriptPubKey**: bao gồm hash của public key của người nhận, cùng với các opcode như OP_DUP, OP_HASH160, OP_EQUALVERIFY và OP_CHECKSIG.
- **scriptSig**: được sử dụng khi người nhận muốn "mở khóa" và sử dụng được số Bitcoin trong giao dịch.

2.2.2 Nguyên lý hoạt động và Quy trình xác minh giao dịch của P2PKH

Nguyên lý hoạt động của P2PKH:

- **Tạo địa chỉ Bitcoin**: Khi tạo một địa chỉ Bitcoin mới, một cặp private key và public key được sinh ra. Địa chỉ Bitcoin sau đó được tạo ra từ việc hash public key.
- **Gửi Bitcoin**: Khi một người dùng muốn thực hiện một giao dịch gửi một số Bitcoin đến một người khác, họ sẽ tạo một giao dịch và sử dụng địa chỉ Bitcoin của người nhận làm đích đến. Giao dịch này chứa scriptPubKey, bao gồm địa chỉ Bitcoin của người nhận và các opcode như OP_DUP, OP_HASH160, OP_EQUALVERIFY và OP_CHECKSIG.
- **Mở khóa giao dịch**: Số Bitcoin này được "khóa" bằng cách sử dụng scriptPubKey trong giao dịch. Người nhận phải cung cấp được public key và chữ ký số hợp lệ để "mở khóa" và sử dụng số Bitcoin này.

Quy trình xác minh giao dịch P2PKH:

1. **Kết hợp scriptSig và scriptPubKey**: Khi một giao dịch P2PKH được tạo, người nhận phải cung cấp scriptSig để xác minh giao dịch này, bao gồm public key và chữ ký số hợp lệ. Trong quá trình xác minh, scriptSig này được kết hợp với scriptPubKey trong giao dịch để đảm bảo tính chính xác và bảo mật.
2. **Thực hiện các Bước Xác Minh**: Giao dịch được xác minh theo các bước sau:
    - **Thêm vào Stack**: scriptSig được đưa vào stack trước, sau đó là scriptPubKey.
    - **Duyệt từng Bước**: Các bước duyệt bao gồm sao chép, băm, kiểm tra độ bằng nhau và xác minh chữ ký bằng các opcode như OP_DUP, OP_HASH160, OP_EQUALVERIFY và OP_CHECKSIG được sử dụng trong quá trình này.
3. **Kiểm tra độ bằng nhau và xác minh chữ ký**:
    - **Băm Public key**: Public key trong scriptSig được băm và so sánh với hash public key trong scriptPubKey (địa chỉ Bitcoin của người nhận).

- **Xác minh chữ ký**: Chữ ký được xác minh với public key. Nếu chữ ký hợp lệ, giao dịch được coi là hợp lệ.

4. **Chấp nhận hoặc từ chối giao dịch**: Nếu tất cả các bước trên được thực hiện thành công và chữ ký số hợp lệ, giao dịch được mạng lưới Bitcoin chấp nhận. Nếu không, giao dịch sẽ bị từ chối.

Stack	Script	Description
Empty.	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	scriptSig and scriptPubKey are combined.
<sig> <pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Constants are added to the stack.
<sig> <pubKey> <pubKey>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is duplicated.
<sig> <pubKey> <pubHashA>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is hashed.
<sig> <pubKey> <pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG	Constant added.
<sig> <pubKey>	OP_CHECKSIG	Equality is checked between the top two stack items.
true	Empty.	Signature is checked for top two stack items.

Hình 4: Bảng quy trình xác minh giao dịch P2PKH [2]

2.2.3 Ưu và nhược điểm của P2PKH

Ưu Điểm của P2PKH:

- **Dễ nhận biết lỗi**: Địa chỉ Bitcoin trong P2PKH bao gồm một checksum, giúp phát hiện lỗi và ngăn chặn giao dịch sai.
- **An toàn:** Cung cấp mức độ bảo mật tốt, vì cần một chữ ký từ private key tương ứng với public key.

Nhược Điểm của P2PKH:

- **Hạn chế về tính linh hoạt:** Không hỗ trợ các cấu hình phức tạp như đa chữ ký hay các điều kiện mở khóa khác ngoài chữ ký đơn.

2.3 Pay-to-Script-Hash script

P2SH không tồn tại dưới dạng script tiêu chuẩn trên mạng Bitcoin cho đến khi BIP 16 được triển khai vào năm 2012. Nó nhanh chóng nhận được sự ủng hộ từ những người dùng muốn có chức năng bổ sung không thể thực hiện được với thanh toán P2PK hoặc P2PKH.

P2SH hoạt động bằng cách băm redeem script thay vì một khóa công khai. Redeem script này xác định cách bitcoin nhận được đến địa chỉ P2SH có thể được chi tiêu trong tương lai.

Ví dụ: một địa chỉ P2SH có thể được sử dụng để tạo một ví đa chữ ký. Trong trường hợp này, script P2SH sẽ chứa danh sách các khóa công khai cần thiết để chi tiêu số tiền. Chỉ khi tất cả các khóa công khai trong danh sách được cung cấp, giao dịch mới có thể được xác nhận.

2.3.1 Cấu trúc P2SH

Cấu trúc của P2SH (Pay to Script Hash) bao gồm hai thành phần chính: scriptPubKey và scriptSig, tương tự như cấu trúc của P2PKH.

Hình 3: Ví dụ cấu trúc P2SH [4]

- **scriptPubKey**: bao gồm một địa chỉ hash được tạo từ một script của người nhận. Các opcode như OP_HASH160 và OP_EQUALVERIFY được sử dụng trong quá trình tạo địa chỉ P2SH.
- **scriptSig**: Được sử dụng khi người nhận muốn "mở khóa" và sử dụng số Bitcoin trong giao dịch.

2.3.2 Nguyên lý hoạt động và Quy trình xác minh giao dịch của P2SH [5]

Nguyên lý hoạt động của P2SH:

- **Tạo kịch bản giao dịch**: Đầu tiên, người tạo giao dịch xây dựng một kịch bản giao dịch. Kịch bản này có thể chứa yêu cầu về việc cần nhiều chữ ký (multi-signature), hoặc các điều kiện đặc biệt khác để giao dịch được xác nhận.
- **Hash kịch bản giao dịch**: Kịch bản được hash để tạo ra một giá trị hash duy nhất. Điều này giúp ẩn đi các chi tiết của kịch bản, đồng thời tạo ra một định danh ngắn gọn hơn cho nó.
- **Tạo địa chỉ P2SH**: Giá trị hash của kịch bản giao dịch sau đó được sử dụng để tạo ra một địa chỉ Bitcoin đặc biệt, gọi là địa chỉ P2SH.

- **Gửi Bitcoin**: Khi một người dùng muốn thực hiện một giao dịch gửi một số Bitcoin đến một người khác, họ sẽ tạo một giao dịch và chứa scriptPubKey là mã hash của script.
- **Mở khóa giao dịch P2SH**: Khi người nhận muốn chi tiêu số Bitcoin đã được khóa bởi P2SH, họ phải cung cấp script gốc và dữ liệu tương ứng (redeem script). Điều này được thực hiện bằng cách thêm vào giao dịch một scriptSig, chứa script thực sự và chữ ký số hợp lệ để "mở khóa" và sử dụng số Bitcoin này.

Quy trình xác minh giao dịch P2PKH:

- **Kết hợp scriptSig và scriptPubKey**: Khi một giao dịch P2SH được tạo, người nhận cung cấp một scriptSig chứa dữ liệu cần thiết để giải mã script được hash (redeem script). ScriptSig này được kết hợp với scriptPubKey trong giao dịch để xác minh tính chính xác và giải mã giao dịch.
- **Thực hiện các Bước Xác Minh**: Dữ liệu từ scriptSig và scriptPubKey được đưa vào một stack theo thứ tự. Các bước xác minh bao gồm duyệt stack, thực hiện các opcode (OP_DUP, OP_HASH160, OP_EQUALVERIFY, OP_CHECKSIG).
- **Băm và Xác Minh Script**: Bước này thực hiện việc băm (hash) scriptSig để tạo ra một giá trị hash. Hash script này được so sánh với scriptPubKey (địa chỉ P2SH) để kiểm tra độ bằng nhau. Nếu không khớp, giao dịch sẽ bị từ chối.
- **Thực Hiện Redeem Script**: Nếu các bước trước đó đều thành công, redeem script được dùng để xác minh tính hợp lệ của giao dịch. Mã script có thể chứa các điều kiện phức tạp như yêu cầu chữ ký số từ nhiều khóa riêng biệt.
- **Chấp Nhận hoặc Từ Chối Giao Dịch**: Nếu tất cả các bước xác minh thành công và mã script thỏa mãn điều kiện, giao dịch được mạng lưới Bitcoin chấp nhận. Ngược lại, nếu có bất kỳ lỗi nào trong quá trình này, giao dịch sẽ bị từ chối.

2.3.3 Ưu điểm và hạn chế của thanh toán P2SH

Bởi vì P2SH được phát triển sau vài năm so với Bitcoin Script ban đầu, các nhà phát triển đã có cơ hội xây dựng trên những ưu điểm và giải quyết những hạn chế của các thiết kế trước đó. Kết quả là P2SH có nhiều ưu điểm hơn so với những hạn chế.

Ưu điểm

- **Tăng cường bảo mật:** Địa chỉ đa chữ ký trong P2SH yêu cầu nhiều chữ ký từ các khóa khác nhau, phân chia quyền truy cập giữa nhiều bên, giúp giảm rủi ro mất mát hoặc trộm cắp tiền mã hóa, tăng cường an toàn bảo mật đáng kể so với P2PKH.
- **Linh hoạt và đa dạng:** Cho phép tùy chỉnh các loại giao dịch phức tạp, bao gồm những cấu hình đa chữ ký như "2-of-2", "2-of-3" hoặc "3-of-5".

Hạn chế

- **Phức tạp**: P2SH ban đầu được phát triển để cung cấp giải pháp cho ví đa chữ ký. Tuy nhiên, hầu hết người dùng cuối sẽ không có nhu cầu về các script phức tạp, do đó việc áp dụng P2SH có phần hạn chế.

3 Cài đặt

Chương này trình bày về cách cài đặt code Python để tạo địa chỉ P2PKH và 2-of-2 multisig cũng như tạo giao dịch từ 2 địa chỉ này và broadcast chúng lên mạng Bitcoin testnet. Nhóm cài đặt các quy trình bên dưới bằng 2 thư viện Python là bitcoinlib [6] và python-bitcoinlib [7]. Cài đặt cũng dựa trên mẫu mà các thư viện này cung cấp.

3.1 Tạo địa chỉ P2PKH

Quy trình của đoạn code để tạo một cặp private key và publickey cùng với một địa chỉ P2PKH trên Bitcoin tesnet:

- Chuyển môi trường sang Testnet
- Tạo private key: tạo một private key từ một chuỗi byte ngẫu nhiên dài 32 byte.
- Tạo public key và địa chỉ Bitcoin từ private key. Địa chỉ này tuân theo định dạng Pay-to-PubKey-Hash (P2PKH).

3.2 Tạo giao dịch cho địa chỉ P2PKH

Quy trình của đoạn code để tạo và gửi một giao dịch Bitcoin từ địa chỉ P2PKH.

- Đầu vào: private key, transaction ID (txid) chứa output để sử dụng, output index, số lượng Bitcoin gửi đi (sent_amount), số lượng Bitcoin giữ lại (kept_amount), và địa chỉ Bitcoin nhận (destination address).
- Chuyển môi trường sang Testnet.
- Tạo và Chuẩn Bị các Thành Phần Giao Dịch:
    - Tạo private key và public key từ input.
    - Tạo transaction input sử dụng txid và output index được cung cấp.

    - Tạo script public key dựa trên public key, sử dụng các opcodes như OP_DUP, OP_HASH160, Hash160, OP_EQUALVERIFY, và OP_CHECKSIG.
    - Tạo hai transaction outputs, một cho người nhận (destination address) và một để trả lại số tiền còn lại (kept_amount) cho người gửi.
- Ký và Tạo Giao Dịch:
    - Giao dịch được tạo với input và output đã chuẩn bị.
    - Tạo chữ ký kỹ thuật số cho giao dịch.
    - Thêm chữ ký và public key vào scriptSig của transaction input.
- Xác Minh tính hợp lệ của scriptSig so với scriptPubKey.
- Gửi Giao Dịch:
    - Giao dịch được serialize dưới dạng hex.
    - Giao dịch được broadcast lên Bitcoin testnet.

Tóm lại, quy trình này tạo một giao dịch Bitcoin trên mạng Testnet, chuyển một số lượng Bitcoin xác định đến một địa chỉ cụ thể, và trả lại một số lượng cho người gửi, phần còn lại là phí giao dịch. Nó sử dụng các opcodes của script Bitcoin để xác nhận và xử lý giao dịch.

3.3 Tạo địa chỉ 2-of-2 multisig (P2SH).

Quy trình tạo một địa chỉ 2-of-2 multisig:

- Chuyển môi trường sang Testnet
- Tạo hai private key từ hai chuỗi byte ngẫu nhiên, mỗi chuỗi dài 32 byte.
- Tạo public key từ các private key:
- Tạo redeem script cho một địa chỉ multisig 2-of-2.
    - Script bao gồm `OP_2`, hai public keys, một lần nữa `OP_2`, và `OP_CHECKMULTISIG`. `OP_2` đại diện cho yêu cầu hai trong số các khóa được cung cấp phải ký giao dịch. `OP_CHECKMULTISIG` thực hiện kiểm tra chữ ký.
- Tạo địa chỉ P2SH từ redeem script.

3.4 Tạo giao dịch cho địa chỉ P2SH

Quy trình tạo và gửi một giao dịch Bitcoin từ địa chỉ P2SH trên mạng Testnet, sử dụng hai khóa cá nhân cho việc xác nhận giao dịch:

- Đầu Vào: hai private keys, transaction ID (txid), output index, số lượng Bitcoin gửi đi (sent_amount), số lượng Bitcoin giữ lại (kept_amount), và địa chỉ Bitcoin nhận (destination address).
- Chọn Mạng Testnet.
- Tạo Khóa và Redeem Script:
    - Tạo hai đối tượng CBitcoinSecret từ hai private keys đầu vào, và từ đó tạo ra hai public keys tương ứng.
    - Tạo redeem script cho địa chỉ P2SH sử dụng cả hai public keys và các opcodes OP_2, OP_CHECKMULTISIG.
- Tạo Input và Output cho Giao Dịch:
    - Tạo transaction input sử dụng txid và output index.
    - Tạo hai transaction outputs: một gửi đến địa chỉ đích (destination address) và một trả lại số lượng Bitcoin còn lại cho địa chỉ P2SH.
- Tạo và Ký Giao Dịch:
    - Tạo một giao dịch mới với input và output đã chuẩn bị.
    - Ký giao dịch sử dụng cả hai private keys. Mỗi chữ ký được tạo dựa trên hash của giao dịch và redeem script.
    - ScriptSig của transaction input bao gồm một phần tử rỗng, theo sau là hai chữ ký và redeem script.
- Xác minh scriptSig so với scriptPubKey dựa trên redeem script.
- Gửi Giao Dịch và In Transaction ID:
    - Serialize giao dịch dưới dạng hex.
    - Broadcast giao dịch lên Bitcoin testnet.

Tóm lại, quy trình này tạo một giao dịch Bitcoin P2SH yêu cầu hai chữ ký từ hai khóa cá nhân khác nhau để được xác nhận.

3.5 Thực nghiệm với ví P2PKH

Quá trình thực nghiệm tạo ví P2PKH và tạo giao dịch từ ví P2PKH gồm các bước sau

Tạo 2 ví P2PKH bằng cách chạy 2 lần lệnh sau:

```
$python create_p2pkh_address.py
```

Thu được 2 địa chỉ như sau:

```
Bitcoin Address: mxFy4CMb7gZQpYkdwjQRGscXyRnA5cUMNL
```

```
Bitcoin Address: mwf5ry6ShJqRTpqNjH4EX9D2aES7kB7adG
```

Truy cập https://coinfaucet.eu/en/btc-testnet/ để nhận 0.00312207 BTC từ faucet qua giao dịch có ID 840642a8170825107a5c2329d974f12dd18ce8fd3838c29eeb4a8904f70af205 với output index là 1. Địa chỉ ví nhận là mxFy4CMb7gZQpYkdwjQRGscXyRnA5cUMNL.

Tạo giao dịch có 2 output:

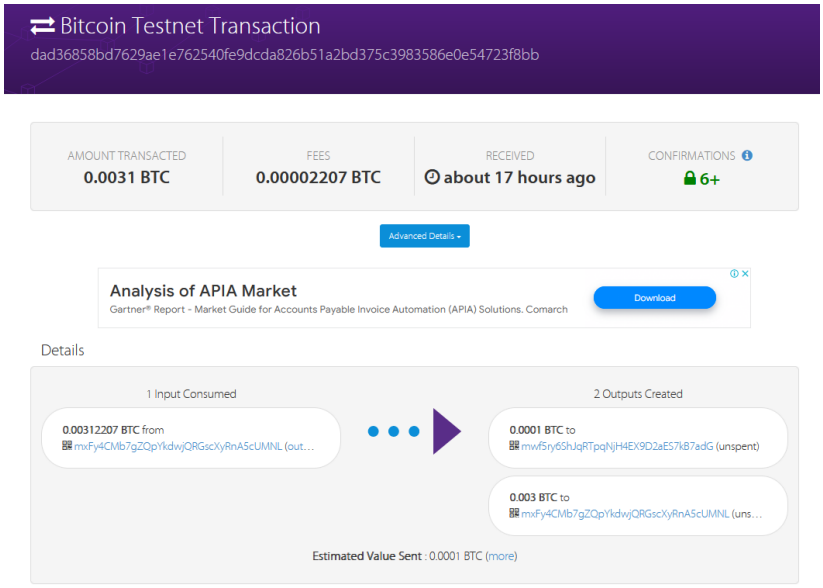
- Gửi 0.0001 BTC tới ví mwf5ry6ShJqRTpqNjH4EX9D2aES7kB7adG.
- Hoàn 0.003 BTC trả về ví mxFy4CMb7gZQpYkdwjQRGscXyRnA5cUMNL.
- 0.00002207 còn lại là phí giao dịch.

```
python p2pkh_script.py \
--private_key=<private key của ví mxFy4CMb7gZQpYkdwjQRGscXyRnA5cUMNL> \
--txid=840642a8170825107a5c2329d974f12dd18ce8fd3838c29eeb4a8904f70af205 \
--output_idx=1 --sent_amount=0.0001 --kept_amount=0.003 \
--destination_address=mwf5ry6ShJqRTpqNjH4EX9D2aES7kB7adG
```

ID của giao dịch là:

dad36858bd7629ae1e762540fe9dcda826b51a2bd375c3983586e0e54723f8bb

Kiểm tra thông tin của giao dịch trên https://live.blockcypher.com để xác thực giao dịch có thành công hay không



Như vậy, giao dịch đã được thực hiện thành công.

3.6 Thực nghiệm với ví P2SH

Quá trình thực nghiệm tạo ví P2SH và tạo giao dịch từ ví P2SH gồm các bước sau

Tạo 1 ví P2SH bằng cách chạy lệnh sau:

```
$python create_multisig_address.py
```

Thu được 1 địa chỉ như sau:

```
Bitcoin Address: 2MsoxohNCkdZegRPXvJJk69SaZXcWUBCL9M
```

Truy cập https://coinfaucet.eu/en/btc-testnet/ để nhận 0.00235237 BTC từ faucet qua giao dịch có ID 4b2763d061c9f9e9a9d5ce4c236a07f6a4a51909c31d9eaf75d6f8db809999d5 với output index là 1 cho địa chỉ P2SH vừa tạo

Tạo giao dịch có 2 output:

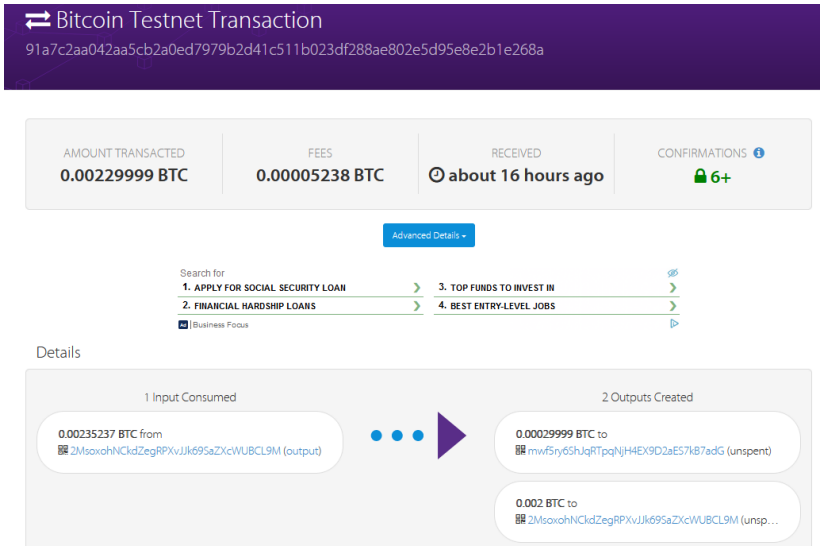
- Gửi 0.0003 BTC tới ví mwf5ry6ShJqRTpqNjH4EX9D2aES7kB7adG
- Hoàn 0.002 BTC trở về ví của chính mình
- 0.00005237 BTC còn lại là phí giao dịch

```
python multisig_script.py

--private_key1=cTyg7TVxxagxrA4pYJHdpiUZ5MeBkn38KmBW36epUjiFfphu4SEk \

--private_key2=cQfws9QgQMwmWtSxCULbTkHkAsjaF9bL2W6DVuMwuGQjTDgHk2ho \

--txid=4b2763d061c9f9e9a9d5ce4c236a07f6a4a51909c31d9eaf75d6f8db809999d5 \

--output_idx=1 --sent_amount=0.0003 --kept_amount=0.002 \

--destination_address=mwf5ry6ShJqRTpqNjH4EX9D2aES7kB7adG
```

ID của giao dịch là:

91a7c2aa042aa5cb2a0ed7979b2d41c511b023df288ae802e5d95e8e2b1e268a

Kiểm tra thông tin của giao dịch trên https://live.blockcypher.com để xác thực giao dịch có thành công hay không



Như vậy, giao dịch đã được thực hiện thành công.

4 Kết luận

Thông qua việc nghiên cứu về Bitcoin Script cùng với 2 loại script là Pay-to-Public-Key-Hash và Pay-to-Script-Hash, nhóm đã có cái nhìn ban đầu về Bitcoin và cách thức vận hành của nó. Để đảm bảo tính xác thực vào bảo mật cho các giao dịch, các giao dịch Bitcoin cần phải được thực hiện thông qua các quy tắc được tuân thủ một cách chặt chẽ.

Quá trình tìm hiểu và cài đặt P2PKH và P2SH cũng đã gặp một vài trở ngại, xong qua đó nhóm cũng đã rút ra được cái bài học liên quan:

- **Hiểu biết kỹ thuật**: Việc nắm vững cơ chế hoạt động của P2PKH và P2SH đòi hỏi sự hiểu biết kỹ thuật về script, public key và private key, cũng như cách thức mà Bitcoin network xác minh và xử lý các giao dịch.
- **Triển khai và debugging**: Quá trình cài đặt script cho cả P2PKH và P2SH gặp phải nhiều khó khăn, đặc biệt trong việc debug và xác minh tính chính xác của script. Mỗi lỗi nhỏ trong script có thể dẫn đến việc giao dịch không được xác nhận.
- **Hiểu biết về an toàn bảo mật**: Việc tìm hiểu về P2PKH và P2SH cũng giúp nhận thức rõ hơn về tầm quan trọng của an toàn bảo mật trong giao dịch Bitcoin.

Bài lab này không chỉ giúp cung cấp kiến thức cơ bản về Bitcoin và cách thức hoạt động của nó, mà còn giúp nhóm thực hành và áp dụng kiến thức này vào thực tế. Thông qua những thách thức trong quá trình thực hiện, nhóm đã học được cách kiểm tra kỹ lưỡng và giải quyết vấn đề một cách có hệ thống.

Tài liệu tham khảo

[1] (2023, June 29). *Bitcoin Script*. https://en.bitcoin.it/wiki/Script

[2] (2023, December 21). *Pay-to-Pubkey Hash.* https://bitcoinwiki.org/wiki/pay-to-pubkey-hash

[3] (2019, January 4). *P2PKH Pay To Pubkey Hash*. https://learnmeabitcoin.com/technical/p2pkh

[4] (2022, August 27). P2SH Pay To Script Hash. https://learnmeabitcoin.com/technical/p2sh

[5] (2023, December 21). Pay-to-Script Hash. https://bitcoinwiki.org/wiki/pay-to-script-hash

[6] (2023, November 24). Bitcoin Library. Github. Retrieved January 1, 2024, from https://github.com/1200wd/bitcoinlib

[7] (2023, December 8). Python-bitcoinlib. Github. Retrieved January 1, 2024, from https://github.com/petertodd/python-bitcoinlib