ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

Môn học: **BLOCKCHAIN: NỀN TẢNG, ỨNG DỤNG VÀ BẢO MẬT**

BÁO CÁO CUỐI KỲ

Đề tài: ZKBRIDGE
CROSS-CHAIN BRIDGE ỨNG DỤNG ZERO-KNOWLEDGE PROOF

Lớp: **NT547.O11.ATCL**

Giảng viên hướng dẫn:

NGUYỄN NGỌC TỰ
TRẦN TUẤN DŨNG
PHAN THẾ DUY

Thành viên:

Nguyễn Mạnh Cường	20520421
Hoàng Văn Anh Đức	20520890

Thành phố Hồ Chí Minh
Ngày 18 tháng 12 năm 2023

MỤC LỤC

MỤC LỤC 1
A. TỔNG QUAN 2
1. Giới thiệu 2
2. Tổng quan đề tài 3
B. PHƯƠNG PHÁP ĐỀ XUẤT 5
1. Mô hình tổng quan 5
2. Các thành phần 8
2.1. Block header relay network 8
2.2. Updater contract 10
2.3. Application contract 11
3. Hệ thống xác minh 12
3.1. deVirgo 12
3.2. Groth16 13
C. TRIỂN KHAI THỰC NGHIỆM 14
1. Vấn đề triển khai 14
2. Axelar Network 14
3. Triển khai 18
D. TRÍCH DẪN 19

BÁO CÁO CHI TIẾT

A.TỔNG QUAN

1. Giới thiệu

- Hiện nay, sự đa dạng trong hệ sinh thái, nơi một loạt các blockchain tồn tại cùng nhau đang phát triển đáng kể. Nhiều blockchain mới cũng xuất hiện với cấu trúc, cơ chế khác nhau và các điểm mạnh riêng trong nhiều lĩnh vực. Tuy nhiên, sự rời rạc của các blockchain riêng biệt trở thành rào chắn cho sự phát triển phổ biến của lĩnh vực blockchain nói chung và từng blockchain nói riêng.
- Cầu nối giữa các chuỗi (cross-chain bridge) là một giải pháp xây dựng cần thiết trong hệ sinh thái đa chuỗi này. Từ đó, hệ sinh thái sẽ có thể phát triển thành một tương lai đa chuỗi nơi các giao thức đa dạng tồn tại cùng nhau, và nhà phát triển cũng như người dùng có thể chọn sử dụng blockchain tốt nhất dựa trên sở thích, chi phí và các tiện ích được cung cấp. Một hệ thống hiệu quả của các cầu nối đối với blockchain sẽ tương tự như cách Internet đã làm cho các mạng truyền thông riêng biệt.
- Các giải pháp hiện tại thường gặp vấn đề về hiệu suất hoặc phụ thuộc vào giả định tin cậy vào các committee (bên trung gian thứ ba), điều này giảm đáng kể độ an toàn. Tài sản được bridge không an toàn bằng tài sản trong nội bộ blockchain, dẫn đến vấn đề phân tích bảo mật của ứng dụng. Committee nhỏ, phân bố đồng thuận không cao, dễ bị thao túng kết quả, đẫn đến nhiều cuộc tấn công lớn đã xảy ra. Hiện tại, vẫn chưa có đề xuất nào hoàn thiện, được công nhận và sử dụng rộng rãi.

2. Tổng quan đề tài

- zkBridge là một cầu nối xuyên chuỗi phi tập trung hiệu quả, không cần giả định tin cậy vào một committee trung gian. Ứng dụng Zero-Knowledge Proof (ZKP), cụ thể là zk-SNARK (Succinct Non-interactive Arguments of Knowledge) cho việc xác minh trao đổi qua bridge. Đồng thời, hỗ trợ nhiều khả năng mở rộng ứng dụng trên bridge bằng việc tách biệt chức năng cơ bản của cầu nối với logic của ứng dụng.
- Lợi ích mang lại, bao gồm:
    - Thứ nhất, tính đúng đắn của thuật toán zk-SNARK đảm bảo an ninh cho cầu nối và không cần yêu cầu bảo mật bổ sung ngoài tính bảo mật của các blockchain cơ bản, không phụ thuộc vào một committee cho vấn đề an ninh.
    - Thứ hai, với zk-SNARK, chuỗi đích có thể xác minh một giao dịch chuyển trạng thái trên chuỗi nguồn hiệu quả hơn nhiều so với việc mã hóa logic đồng thuận, điều đó làm giảm gánh nặng lưu trữ của cầu nối.
    - Thứ ba, bằng việc tách rời cầu nối khỏi logic cụ thể của ứng dụng, zkBridge giúp dễ dàng kích hoạt các ứng dụng bổ sung trên cầu nối.
- Tuy nhiên, việc ứng dụng ZKP vào quá trình tạo proof, xác minh proof cho các giao dịch qua cầu nối gặp phải một số vấn đề về hiệu năng và chi phí. Vì vậy, đề xuất hệ thống xác minh đệ quy 2 lớp, độc lập và có giá trị lợi ích riêng, với cơ sở là zk-SNARK để đạt được cả thời gian tạo proof hợp lý và chi phí xác minh trên chuỗi.
    - Tầng đầu tiên, nhằm mục tiêu tạo proof nhanh chóng, đề xuất deVirgo, một phiên bản phân phối (distributed) của hệ thống xác minh Virgo. deVirgo kết hợp sumcheck phân phối và cam kết đa thức phân phối để đạt

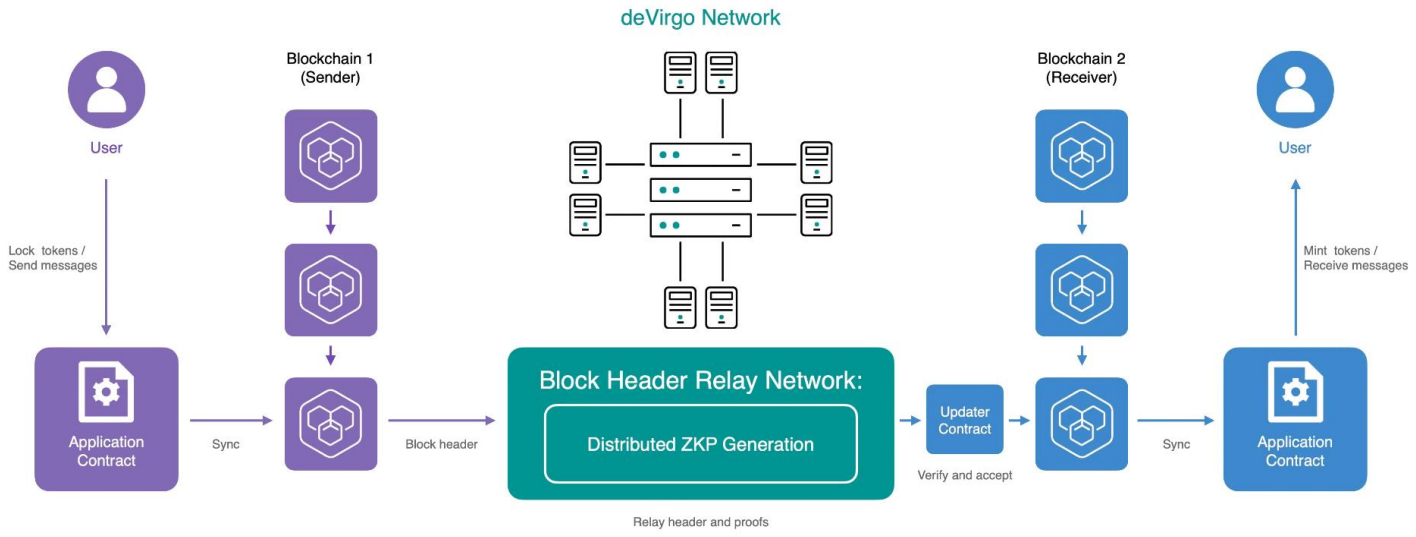
được sự song song tối ưu, qua đó giai đoạn tạo proof được gia tăng đáng kể thông qua việc chạy trên các máy tính phân phối.

- Tầng thứ hai, nhằm mục tiêu giảm chi phí xác minh trên chuỗi, sử dụng Groth16 để xác minh đệ quy rằng proof đã được tạo trước đó bởi deVirgo thực sự xác minh sự hợp lệ của các block header tương ứng.

B. PHƯƠNG PHÁP ĐỀ XUẤT

1. Mô hình tổng quan

- Một hợp đồng thông minh là một chương trình có trạng thái được lưu trữ trên một blockchain. Một cầu nối như zkBridge là một dịch vụ cho phép các hợp đồng thông minh trên các blockchain khác nhau chuyển trạng thái từ một chuỗi sang một chuỗi khác một cách an toàn và có xác minh.
- zkBridge được thiết kế theo cấu trúc modular, qua đó tách logic cụ thể của ứng dụng (xác minh trạng thái hợp đồng thông minh,…) khỏi chức năng cốt lõi của cầu nối (chuyển tiếp block header). zkBridge cũng hỗ trợ và yêu cầu blockchain nguồn có hỗ trợ Light client protocol, điều đó cho phép việc đồng bộ trạng thái qua các chuỗi chỉ bằng tiêu đề khối (block header).

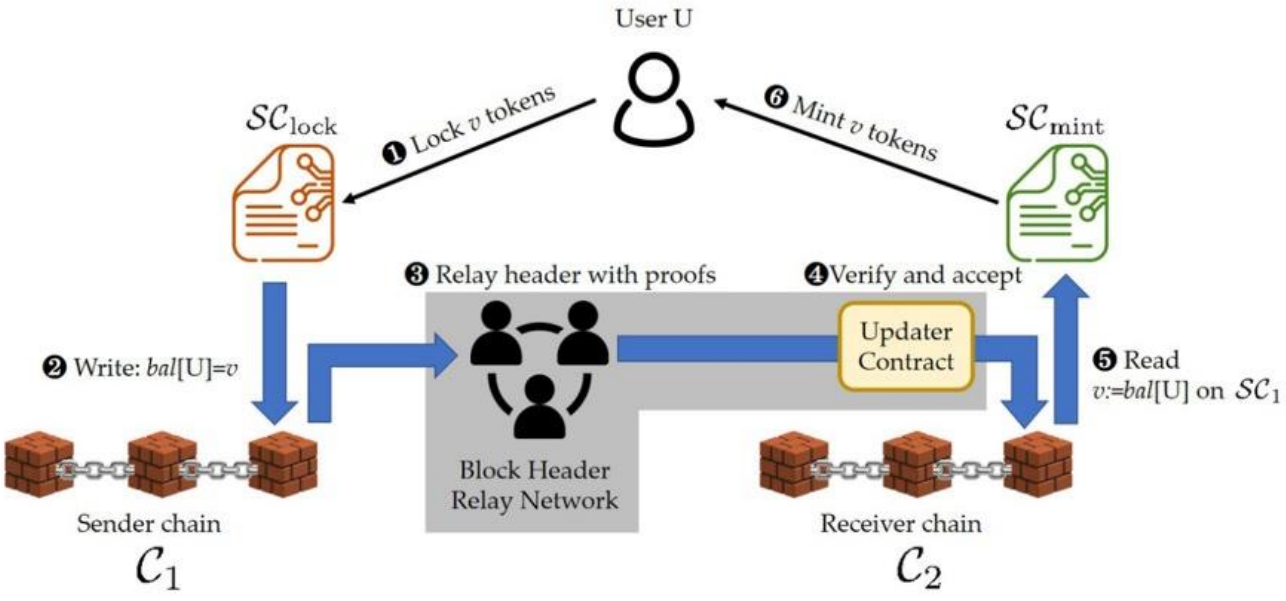


Hình 1: *Cấu trúc zkBridge.*

- Chức năng cầu nối cốt lõi được thực hiện bởi một mạng truyền tiêu đề khối (Block header relay network) bao gồm các nút (node) chuyển tiêu đề khối của Blockchain 1 (C1) cùng với proof độ chính xác (proof) đến một hợp đồng cập

nhật (Updater contract) trên Blockchain 2 (C2) xác minh và chấp nhận proof được gửi bởi các nút truyền tiếp (node trong Block header relay network). Hợp đồng cập nhật duy trì một danh sách các tiêu đề khối gần đây và cập nhật nó sau khi xác minh proof được gửi bởi các nút truyền tiếp; nó mở ra một API đơn giản và không phụ thuộc ứng dụng, từ đó các hợp đồng thông minh ứng dụng có thể lấy tiêu đề khối mới nhất của blockchain gửi và xây dựng logic cụ thể cho ứng dụng dựa trên đó.

- Các ứng dụng triển khai phụ thuộc vào zkBridge thường sẽ triển khai một cặp hợp đồng, một hợp đồng nguồn và một hợp đồng đích trên C1 và C2 tương ứng, gọi là hợp đồng ứng dụng (Application contract). Hợp đồng đích có thể gọi hợp đồng cập nhật để lấy tiêu đề khối của C1, dựa vào đó nó có thể thực hiện các nhiệm vụ cụ thể của ứng dụng. Tùy thuộc vào ứng dụng, hợp đồng đích cũng có thể cần một người dùng hoặc bên thứ ba để cung cấp xác minh cụ thể cho ứng dụng, chẳng hạn như Merkle proof cho trạng thái hợp đồng thông minh.



Hình 2: *Ví dụ về zkBridge.*

- Ví dụ, **Hình 2** cho thấy quy trình chuyển token giữa các chuỗi, là một trường hợp ứng dụng phổ biến của các cầu nối, được thực hiện bởi zkBridge.
    - Giả sử một người dùng U muốn thực hiện giao dịch tài sản (token) mà họ sở hữu trên blockchain C1 trên một sàn giao dịch đặt trên blockchain C2 (có thể vì C2 tính phí thấp hơn,…), họ cần chuyển số dư của mình từ C1 sang C2.
    - Một cặp hợp đồng thông minh SClock và SCmint được triển khai trên blockchain C1 và C2 tương ứng. Để chuyển số dư, người dùng khoá v token trong SClock **(Bước 1)** và sau đó yêu cầu v token được phát hành bởi SCmint. Để đảm bảo tính bảo mật, SCmint chỉ nên phát hành token mới nếu và chỉ nếu người dùng đã khoá token trên C1. Điều này đòi hỏi SCmint đọc được trạng thái của SClock (số dư của U, **Bước 2**) từ một blockchain khác, điều mà nó không thể làm trực tiếp.
    - zkBridge cho phép thục hiện điều này bằng cách truyền tiêu đề khối của C1 đến C2 cùng với proof **(Bước 3 và 4)**. SCmint có thể lấy tiêu đề khối từ API hợp đồng cập nhật, kiểm tra rằng số dư của người dùng U thực sự là v **(Bước 5)**, và chỉ sau đó mới phát hành v token **(Bước 6)**.
- Ngoài chuyển token giữa các chuỗi, zkBridge cũng có thể hỗ trợ nhiều ứng dụng khác nhau như truyền tin nhắn, dữ liệu, NFT,…

2. Các thành phần

Một cầu nối zkBridge bao gồm ba thành phần: mạng truyền tiêu đề khối, một hợp đồng cập nhật và một hoặc nhiều hợp đồng ứng dụng.

Chú trích các ký hiệu trong các Giao thức:

- **LCS:** Light Client State.
- **blkH:** block Header.
- **r, r–1:** round – thứ tự trạng thái của block header.
- **t:** thứ tự của block header trong headerDAG.

2.1. Block header relay network

Protocol 1 Block header relay network

procedure RELAYNEXTHEADER(LCS_{r-1},$blkH_{r-1}$)

Contact k different full nodes to get the block headers following $blkH_{r-1}$, namely $blkH_r$.

Generate a ZKP π proving

$$LightCC(LCS_{r-1},blkH_{r-1},blkH_r) \rightarrow true.$$

Send (π,$blkH_r$,$blkH_{r-1}$) to the updater contract.

end procedure

Giao thức 1: *Mạng truyền tiêu đề khối.*

- **Giao thức 1** trình bày giao thức chi tiết của mạng truyền tiêu đề khối.
    - Các nút trong mạng truyền tiêu đề khối chạy *RelayNextHeader* với đầu vào là trạng thái hiện tại của hợp đồng cập nhật (LCSr−1, blkHr−1). Định nghĩa chính xác của LCSr−1 (trạng thái Light client) phụ thuộc vào Light client protocol được sử dụng.
    - Nút truyền tiếp sau đó kết nối với các nút đầy đủ (full node) trong C1 và nhận tiêu đề khối blkHr theo blkHr−1.
    - Nút truyền tiếp tạo ra một proof ZKP π xác minh tính chính xác của blkHr, bằng cách cơ bản xác minh rằng blkHr được chấp nhận bởi một light node của C1 liền sau khối blkHr−1.
    - Sau đó, nó gửi (π,blkHr) đến hợp đồng cập nhật trên C2.
- Để tránh lãng phí thời gian xác minh do xung đột nhiều nút (khi nhiều nút truyền tiếp gửi cùng một lúc, chỉ có một proof được chấp nhận), các nút truyền tiếp có thể phối hợp sử dụng các kỹ thuật tiêu chuẩn (gửi theo kiểu round robin,…).
- Để khuyến khích các nút truyền tiếp tiêu đề khối, người xác minh có thể được thưởng phí sau khi xác minh được proof họ tạo. Tuy nhiên, phương pháp chưa đưa ra đề xuất về vấn đề này. Điều kiện tiên quyết của bất kỳ đề xuất khuyến khích nào là đảm bảo rằng các nút độc hại (kẻ tấn công) không thể đánh cắp proof của người khác. Để đạt được điều này, người xác minh sẽ nhúng các định danh của họ (chữ ký số) vào proof.
- Lưu ý rằng mô hình thiết kế này phụ thuộc vào tính bảo mật của bộ kiểm tra Light client của chuỗi nguồn.

2.2. Updater contract

Protocol 2 The updater contract

```
headerDAG := ∅                                   ▷ DAG of headers
LCS := ⊥                                         ▷ light client state
procedure HeaderUpdate(π, blkH_r, blkH_{r−1})
    if blkH_{r−1} ∉ headerDAG then
        return False      ▷ skip if parent block is not in the DAG
    end if
    if π verifies against LCS, blkH_{r−1}, blkH_r then
        Update LCS according to the light client protocol.
        Insert blkH_r into headerDAG.
    end if
end procedure

procedure GetHeader(t)    ▷ t is a unique identifier to a block header
    if t ∉ headerDAG then
        return ⊥                                 ▷ tell the caller to wait
    else
        return headerDAG[t], LCS   ▷ The LCS will help users to determine if t is on a fork.
    end if
end procedure
```

Giao thức 2: *Hợp đồng cập nhật.*

- **Giao thức 2** trình bày giao thức chi tiết của hợp đồng cập nhật.
    - The updater contract duy trì trạng thái nội bộ của Light client, bao gồm danh sách các tiêu đề khối của C1 trong headerDAG.
    - Bao gồm hai hàm (API) được công khai:
        - *HeaderUpdate*: có thể được gọi bởi bất kỳ nút truyền tiêu đề khối nào, nhận tiêu đề khối tiếp theo và một proof (đầu ra của *RelayNextHeader*) làm đầu vào. Nếu blkHr−1 tồn tại trong headerDAG, hợp đồng sẽ tiếp tục kiểm tra proof có xác minh được trạng thái Light client hiện tại (LCS) và blkHr−1, sau đó trạng thái sẽ được cập nhật tương ứng. Vì người gọi của hàm này phải trả phí, các cuộc tấn công DoS được ngăn chặn tự nhiên.
        - *GetHeader*: có thể được gọi bởi các hợp đồng nhận (Application contract) để lấy tiêu đề khối tại t. Các hợp đồng nhận có thể sử dụng tiêu đề khối đã nhận được để hoàn thành xác minh cụ thể cho ứng dụng, có thể với sự giúp đỡ của người dùng hoặc một bên thứ ba.

2.3. Application contract

- zkBridge có thiết kế modular với việc hợp đồng cập nhật không phụ thuộc vào ứng dụng. Do đó, tùy thuộc vào hợp đồng ứng dụng SC1 và SC2 quyết định thông tin cần truyền qua cầu nối là gì.
- Việc xác minh trong SC2 có thể yêu cầu một xác minh Merkle proof cho lá của Trie Tree trạng thái (tại t) tương ứng với địa chỉ K. Hợp đồng nhận có thể nhận blkHt từ hợp đồng cập nhật bằng cách gọi hàm GetHeader(t). Sau đó, nó có thể xác minh nội dung so với Merkle root trong blkHt.

- Xác minh bổ sung trên tùy thuộc vào từng ứng dụng và thường được người dùng của SC2, một bên thứ ba hoặc người phát triển/quản trị của SC2 cung cấp.

3. Hệ thống xác minh

Phần đòi hỏi tính toán nhiều nhất của zkBridge là việc tạo ra proof ZKP mà các nút chuyển tiếp phải thực hiện cho mỗi khối và việc xác minh proof đó trên chuỗi.

3.1. deVirgo

- Để kiểm chứng tính chính xác của kết quả tính toán sử dụng một zk-SNARK, trước hết cần phải biểu diễn tính toán đó dưới dạng một mạch số học (arithmetic circuit). Các thành phần được sử dụng bởi các blockchain trong thế giới thực không dễ dàng biểu diễn dưới dạng mạch số học, dẫn đến thời gian tạo proof ít nhất là tuyến tính và trong thực tế là quá đắt đỏ. Để làm cho mô hình đề xuất zkBridge thực tế, phải giảm thời gian tạo proof.
- Quan sát trong ZKP, để xác minh nhiều chữ ký, mạch được tạo thành từ nhiều bản sao của một mạch con. Tận dụng cấu trúc đặc biệt này, có thể phân phối quá trình tạo proof trên nhiều máy chủ. Đề xuất một giao thức ZKP phân tán mới mang tên deVirgo, một phiên bản nâng cấp của giao thức Virgo (zk-SNARK), một trong những hệ thống ZKP nhanh nhất (về thời gian xác minh) mà không cần thiết lập tin cậy.
- Với deVirgo, có thể tăng tốc quá trình tạo proof trong zkBridge một cách tuyến tính hoàn hảo, nghĩa là thời gian tạo proof có thể giảm đi một hệ số M nếu quá trình tạo proof được phân phối trên M máy tính. Giao thức này có giá trị độc lập và có thể hữu ích trong các tình huống khác.

3.2. Groth16

- Mặc dù deVirgo đã giảm đáng kể thời gian tạo proof, việc xác minh deVirgo proof trên chuỗi, đặc biệt là đối với các mạch số có hàng tỷ cổng trong zkBridge, có thể tốn nhiều chi phí trong hợp đồng thông minh (hợp đồng cập nhật) nơi tài nguyên tính toán rất hạn chế.
- Để giảm kích thước proof và chi phí xác minh (tính toán và lưu trữ), cần đánh đổi thời gian tạo proof tăng lên một chút. Đề xuất xác minh đệ quy tính chính xác của một deVirgo proof (có thể rất lớn) bằng một zk-SNARK thân thiện với hợp đồng thông minh để có được một proof nhỏ và hiệu quả cho bên xác minh, được gọi là Groth16.
- Groth16 prover tạo ra proof có kích thước hằng số, dễ xác minh bởi một hợp đồng thông minh trên một blockchain EVM. Nhấn mạnh rằng không thể sử dụng Groth16 để tạo ra toàn bộ zkBridge proof vì các mạch cần thiết trong zkBridge quá lớn cho Groth16 prover.

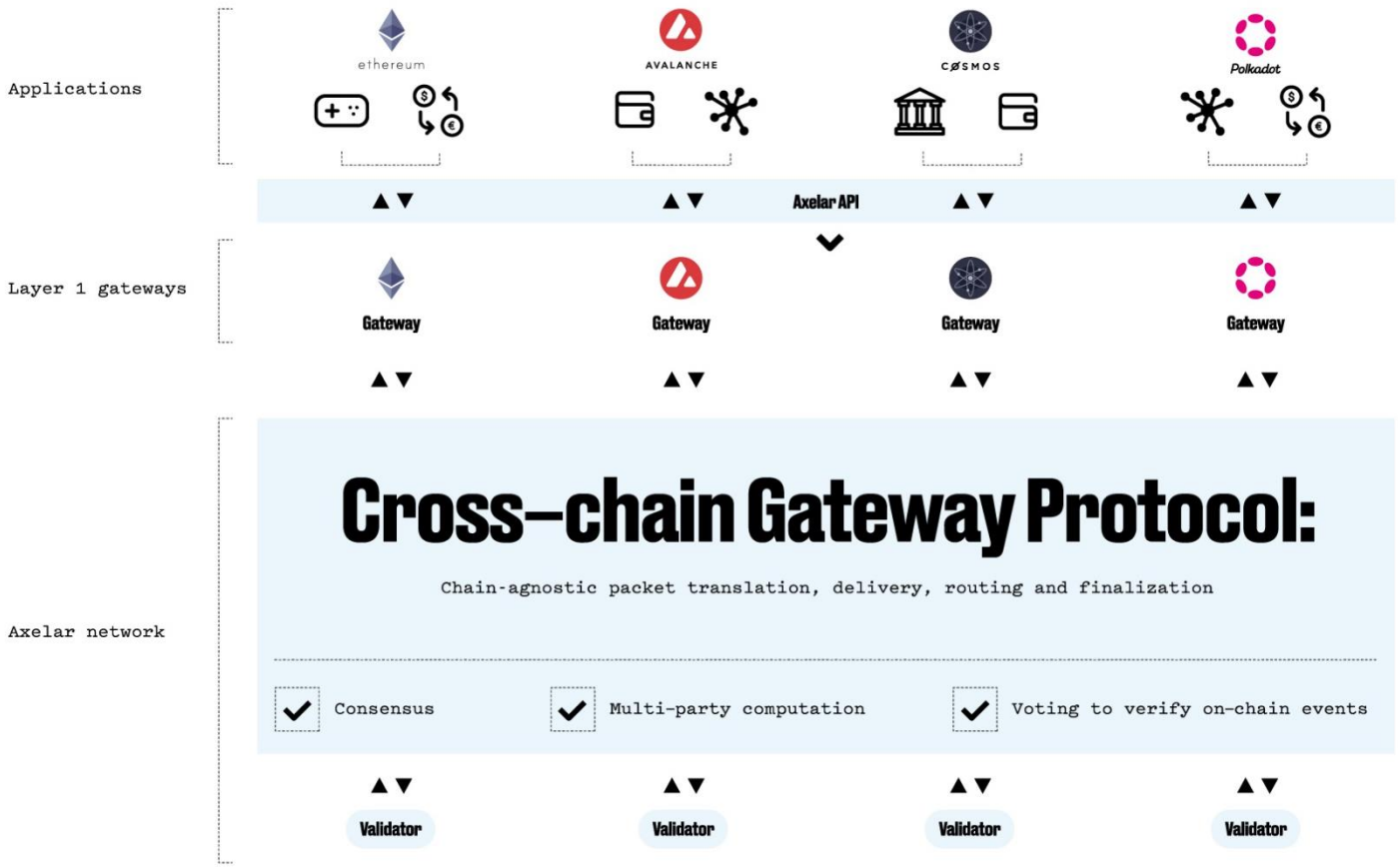
C.TRIỂN KHAI THỰC NGHIỆM

1. Vấn đề triển khai

- Vì nhóm không thể thực hiện triển khai được mô hình tương từ phương pháp đề xuất của bài báo chính, nên nhóm sẽ thực hiện triển khai một mạng Cross-chain Bridge tương tự là Axelar Network.
- Về điểm khác biệt, Axelar không xác minh bằng ZKP mà sử dụng cơ chế đồng thuận Proof-of-Stake. Cấu trúc cũng sẽ khác với việc không phân tách Block header relay network và Updater contract ra riêng.

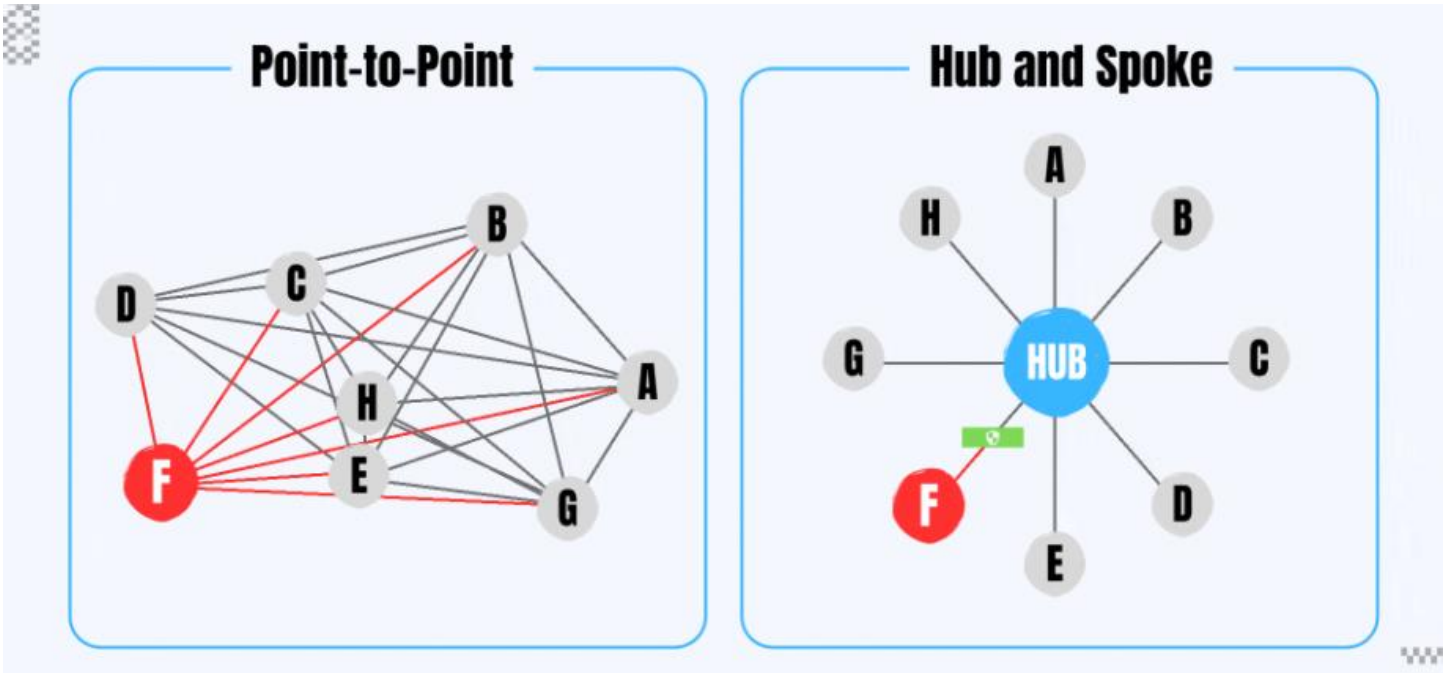
2. Axelar Network

- Axelar Network là một blockchain kết nối các blockchain, sử dụng cơ chế đồng thuận Proof-of-Stake(PoS) và được xây dựng dựa trên bộ công cụ Cosmos SDK. Dự án cung cấp cơ sở hạ tầng hỗ trợ các blockchain tương tác với nhau. Đồng thời, các lập trình viên có thể xây dựng các ứng dụng phi tập trung (dApp) tương tác đa chuỗi nhờ công nghệ của Axelar Network.
- Axelar Network đóng vai trò là bên thứ ba để xác thực độ tin cậy của các giao dịch đa chuỗi, giúp đơn giản hóa quá trình chia sẻ và truyền thông tin (dữ liệu) cho cả người dùng và dApp.



Hình 3: *Cấu trúc Axelar Network*

- Axelar Network sử dụng mô hình Hub-and-Spoke để xây dựng cơ sở hạ tầng, nhằm đảm bảo khả năng tương thích và tương tác giữa các blockchain với nhau. Hub-and-Spoke là mô hình bao gồm một "hub" chính và các "spoke" kết nối với hub. Axelar Chain là "hub", các blockchain khác là "spoke". Khi một blockchain kết nối với Axelar, nó có thể kết nối với những mạng lưới khác.
- Thông qua mô hình trên, Axelar Network cho phép người dùng thực hiện giao dịch đa chuỗi mà vẫn đảm bảo tính bảo mật.



Hình 4: *Mô hình Hub and Spoke.*

- Axelar Network đã phát triển được một số ứng dụng dựa trên công nghệ của mình, bao gồm: Satellite, General Message Passing, Axelar Virtual Machine.
- Axelar Network có native token là AXL. Bên cạnh AXL token, Axelar Network cũng phát hành token wAXL, có thể hoạt động trên tất cả các chuỗi tương thích với EVM.
- Mạng Axelar có ba thành phần chính:
    - Mạng lưới phi tập trung: Đầu tiên là mạng lưới phi tập trung, được hỗ trợ bởi một bộ xác thực động chịu trách nhiệm duy trì mạng và thực hiện các giao dịch. Các trình xác thực chạy giao thức cổng chuỗi chéo, đây là lớp phủ mật mã nhiều bên nằm trên các chuỗi khối Layer 1. Nó chịu trách nhiệm thực hiện các hoạt động đọc và ghi vào gateway được triển khai trên các chuỗi bên ngoài được kết nối, áp dụng sự đồng thuận PoS để xác minh các sự kiện trên các chuỗi đó.

- Hợp đồng thông minh của Gateway: gateway là các cài đặt trên các chuỗi khối được kết nối. Trên chuỗi EVM, chúng là hợp đồng thông minh. Trình xác thực giám sát gateway để phát hiện các giao dịch đến mà trình xác thực sẽ đọc. Sau đó nó sẽ thống nhất về tính hợp lệ của giao dịch đó. Sau khi đồng ý, nó gửi thông tin tới gateway của chuỗi đích để thực hiện giao dịch chuỗi chéo. Trình xác nhận và gateway tạo nên lớp cơ sở hạ tầng cốt lõi.
- API và công cụ dành cho nhà phát triển: Đứng trên các trình xác thực và gateway là các API và SDK (thư viện và công cụ cho phép các nhà phát triển truy cập mạng Axelar một cách dễ dàng). Đây là lớp phát triển ứng dụng mà các nhà phát triển sẽ sử dụng để kết hợp trên hai chuỗi bất kỳ trong một bước nhảy duy nhất, bổ sung khả năng tương tác phổ quát cho các chuỗi khối và ứng dụng của họ.

- Một số điểm nổi bật của Axelar Network:
  - Phi tập trung: Axelar Network triển khai cross-chain trên bất kỳ chain nào kể cả các mạng lưới không có Smart Contract. Tất cả các dữ liệu cập nhật, giao thức,... đều được thực hiện một cách phi tập trung.
  - Cross-chain Communication: Giao thức có nhiều công cụ và các tính năng như API, GMP…giúp tương tác giữa các chuỗi trở nên thuận tiện dễ dàng hơn.
  - Một số ứng dụng khác trong tương lai: Ngoài các ứng dụng đã được phát triển, Axelar Network đang lên kế hoạch mở rộng các ứng dụng khác như: vay và cho vay trong thị trường DeFi, tài sản tổng hợp đa chuỗi (cross-chain synthetic asset), quyền quản trị đa chuỗi (cross-chain governance)...

3. Triển khai

- Triển khai thực hiện giả lập 5 mạng blockchain: Moonbeam, Avalanche, Fantom, Ethereum, và Polygon. Tất cả được triển khai local trên máu chủ cá nhân.
- Các thao tác được thực hiện giữa các blockchain bao gồm: gửi tin nhắn, gửi token, gửi NFT, tạo token xuyên chuỗi,…

D.TRÍCH DẪN

- Tiancheng Xie, Jiaheng Zhang, Zerui Cheng, Fan Zhang, Yupeng Zhang, Yongzheng Jia, Dan Boneh, and Dawn Song. 2022. ZkBridge: Trustless Cross-chain Bridges Made Practical. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS'22). Association for Computing Machinery, New York, NY, USA, 3003–3017. https://doi.org/10.1145/3548606.3560652
- Axelar Network: https://axelar.network/
- Mã nguồn: https://github.com/axelarnetwork/axelar-examples