МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В.Н.Каразіна
Факультет математики і інформатики
Кафедра теоретичної та прикладної інформатики

Кваліфікаційна робота

бакалавр

на тему
“Застосування доказів з нульовим розголошенням у cosmos-based мережах”

Виконав: студент 4 курсу, групи МФ-41
спеціальність 122 «Комп’ютерні науки»
освітня програма «Теоретична і прикладна інформатика»
Кудрявцев Д. Ю.
Керівник: к. т. н., доцент Меняйлов Є.С.
Рецензент:
_____

Харків – 2024 року

Зміст

Перелік позначень та скорочень

PoS	-	Proof-of-Stake
BFT	-	Byzantine Fault Tolerance
PBFT	-	Practical Byzantine Fault Tolerance
ZKP	-	Zero-Knowledge Proof
TPS	-	Transactions Per Second
VC	-	Verifiable Credential
EVM	-	Ethereum Virtual Machine
SDK	-	Software Development Kit
DeFi	-	Decentralized Finance
DAO	-	Decentralized Autonomous Organization
NFT	-	Non-Fungible Token
API	-	Application Programming Interface
IBC	-	Inter-Blockchain Communication
RPC	-	Remote Procedure Call
EOA	-	Externally Owned Account
ОС	-	операційна система
ABI	-	Application Binary Interface
KYC	-	Know Your Customer
AML	-	Anti-Money Laundering
DID	-	Decentralized identifier
ППЗ	-	пам’ять постійного зберігання
Tx	-	транзакція

Вступ

1.1 Формулювання задачі та обґрунтування актуальності теми

У сучасному світі, коли цифрові технології проникають у всі сфери людського життя, питання конфіденційності та захисту даних стають все більш актуальними. Це стосується як повсякденного життя, будь-яких сайтів в інтернеті, так і криптовалютних та блокчейн мереж, які відіграють важливу роль у цифровій економіці. Однак існуючі підходи до забезпечення конфіденційності у таких мережах мають свої обмеження, зокрема, пов'язані з можливістю аналізу транзакцій та витоком інформації про користувачів. Що стосується повсякденного життя, ми розголошуємо багато зайвої інформації, якої від нас не вимагають та яка може бути використана зловмисниками проти нас самих.

Наприклад, під час повсякденних покупок у магазині у Вас можуть попросити пред’явити паспорт для підтвердження віку, у цей момент ми просто показуємо свій фізичний паспорт або електронний у застосунку “Дія”. Тим самим ми надаємо усю нашу інформацію, наявну в паспорті, продавцю, а саме: дату народження, номер паспорту, серію, місце реєстрації, дату видачі документу та дату, до якої він дійсний, хоча касиру треба знати лише, що паспорт дійсний і Вам більше 18 років.

Щодо сайтів в інтернеті та ж ситуація, але крім одноразового перегляду паспорта або будь-яких інших документів, власники або розробники сайту можуть зберігати вашу персональну інформацію на будь-який термін.

У блокчейн мережах надання персональної інформації та документів взагалі не використовується, бо тоді доступ до Вашої інформації буде взагалі у всіх користувачів. У випадку приватних блокчейн мереж іноді

така практика використовується, інформація буде недоступна звичайним користувачам, але все одно будет доступ у власників та її розробників.

Вищезазначені проблеми можуть бути повністю або частково вирішені за допомогою доказів з нульовим розголошенням та можуть забезпечити конфіденційність персональної інформації або хоча б обмежити об’єм її розголошення.

Мета роботи:

1) Дослідити існуючі імплементації доказів з нульовим розголошенням.
2) Дослідити cosmos-based мережі та їх переваги у порівнянні з іншими блокчейн мережами.
3) Інтегрувати, пристосувати та протестувати існуючі рішення у власній cosmos-based мережі.
4) Дослідити можливості подальшого розвитку та перспективи доказів з нульовим розголошенням у cosmos-based мережах.
5) Дослідити use cases доказів з нульовим розголошенням у cosmos-based блокчейн мережах та у реальному світі.

1.2 Огляд існуючих технологій та проектів, які використовують докази з нульовим розголошенням

Докази з нульовим розголошенням (ZKPs) — це криптографічні протоколи, які дозволяють одній стороні довести іншій, що їм відома певна інформація або що значення або твердження правдиві, не розкриваючи жодної додаткової інформації. Деякі з найпоширеніших імплементацій включають:

1. **zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge)**: Це одна з найбільш відомих імплементацій ZKP, яка дозволяє довести, що певні дані є вірними без розголошення конкретної інформації. Ця імплементація забезпечує компактність та ефективність, що робить її ідеальною для застосування в блокчейні.
2. **Bulletproofs**: Це імплементація, яка дозволяє довести правильність певної операції, такої як додавання чи множення, без розголошення конкретних даних. Bulletproofs є ефективними та масштабованими, що робить їх придатними для застосування у різних криптографічних протоколах.
3. **zk-STARKs (Zero-Knowledge Scalable Transparent Arguments of Knowledge)**: Ця імплементація ZKP дозволяє доводити правильність деяких операцій у розподіленому середовищі без необхідності довіри до довіреного органу. Завдяки високій масштабованості та транспарентності, zk-STARKs потенційно можуть бути використані в різних застосуваннях, включаючи блокчейн-технології.
4. **zk-Rollups**: Ця імплементація використовується для масштабування транзакцій у блокчейні, дозволяючи проводити багато транзакцій поза ланцюжком(off-chain) та підтверджувати їхню коректність без

необхідності перевірки кожної транзакції. Завдяки ZKPs, zk-Rollups забезпечують конфіденційність та ефективність масштабування.

Ця концепція була широко прийнята в просторі блокчейну, що породило численні проекти, які реалізують ZKP різними способами. Нижче наведено кілька ключових проектів, які використовують вищезазначені імплементації ZKP:

1. **Polygon ID(zk-SNARKs)**[1]: Це децентралізоване рішення ідентифікації, побудоване на блокчейні Polygon. Система дозволяє користувачам генерувати облікові дані на основі ZKP, які можуть бути перевірені іншими без розкриття основної інформації. Це дає змогу підтвердити ідентифікаційні атрибути, такі як вік або статус членства, без надання конфіденційних даних.
2. **StarkNet(zk-STARKS)**[2]: Рішення для масштабування рівня 2(layer-2) для Ethereum. Він використовує технологію ZKP, щоб забезпечити масштабовану та безпечну обробку транзакцій. StarkNet об’єднує кілька транзакцій в один ZKP, який потім перевіряється в блокчейні Ethereum, що значно знижує витрати на транзакції та покращує пропускну здатність. Забезпечує високий рівень безпеки, дозволяючи прозору публічну перевірку доказів без необхідності довірених налаштувань (trusted setups).
3. **zkSync(zk-Rollups)**[3]: Це ще одне рішення для масштабування рівня 2(layer-2) для Ethereum. Він використовує zk-Rollups для об’єднання кількох транзакцій в одне підтвердження, яке потім перевіряється в основній мережі Ethereum. Цей підхід дозволяє zkSync пропонувати швидкі та недорогі транзакції, зберігаючи безпеку блокчейну. Популярний у просторі DeFi завдяки своїм ефективним і безпечним можливостям обробки транзакцій.

4. **Monero(Bulletproofs)**[4]: Орієнтована на конфіденційність криптовалюта, інтегрувала Bulletproofs у свій протокол у жовтні 2018 року. Це оновлення замінило попередні кільцеві підписи, що призвело до значно коротших розмірів доказів для конфіденційних транзакцій. Monero використовує Bulletproofs для забезпечення конфіденційних і безпечних транзакцій, гарантуючи, що суми транзакцій залишаються конфіденційними, забезпечуючи ефективну перевірку коректності операцій.

Дослідивши наявні імплементації ZKP та проекти, які їх використовують, було прийняте рішення обрати за основу zk-SNARSKs та Polygon ID, який побудований з використанням цієї імплементації доказів з нульовим розголошенням. Переваги обраної технології та проекту:

- достатня документація з описом технологій, принципів роботи та прикладами використання та створення власних схем ZKP;
- простота використання, мінімальна кількість додаткових компонентів для розгортання у мережі;
- наявність тестового середовища;
- велика аудиторія користувачів та розробників, що полегшує розв’язання проблем та багів.

1.3 Обґрунтування використання cosmos-based мережі та порівняння з іншими

Cosmos-based мережа - це блокчейн мережа, яка побудована на основі технологій і протоколів, розроблених в рамках екосистеми Cosmos, а саме з використанням Cosmos SDK[5] (Software Development Kit).

Cosmos SDK має модульну структуру та містить ряд стандартних модулів, необхідних для побудови будь-якої блокчейн мережі, такі як Auth (зберігає інформацію про акаунти користувачів), Bank (зберігає інформацію про баланси користувачів та дає змогу передавати токени між користувачами), Staking (зберігає інформацію про валідаторів, їх voting power та забезпечує можливість делегувати токени користувачів обраним валідаторам) та багато інших. Також Cosmos SDK надає можливість розробникам створювати власні модулі та надає рекомендації щодо їх архітектури.

Саме завдяки підходам, використаним у Cosmos SDK, усі cosmos-based мережі мають модульну структуру, що дозволяє розробникам значно прискорювати розробку власних мереж - швидко додаючи існуючі модулі у свою розробку та генеруючі стандартизований скелет для побудови власних.

Також Cosmos SDK дозволяє використовувати різні алгоритми досягнення консенсусу “під капотом”. Він надає стандартизований інтерфейс спілкування між Cosmos SDK та двигуном консенсусу. Зазвичай для досягнення консенсусу між валідаторами мережі використовують Tendermint[6] (PBFT - Practical Byzantine Fault Tolerance), CometBFT[7] (PBFT), Twin-Turbo consensus mechanism та інші, що дає змогу обирати розробникам двигун консенсусу у відповідності до потреб продукту.

Таким чином у cosmos-based мережах є чіткий поділ на Consensus level (Tendermint, CometBFT та інші) та Business logic level (Cosmos SDK). Саме це дає розробникам можливість концентруватися на необхідній частині продукту та не розробляти двигун консенсусу або бізнес логіку з нуля.

Після аналізу інших блокчейн мереж, в екосистемі яких можливе використання доказів з нульовим розголошенням, було створено таблицю для порівняння та виявлення їх недоліків і переваг та для визначення платформи для подальшої розробки.

Критерії порівняння	Cosmos-based	Ethereum	Polygon
Консенсус	BFT-based DPoS	PoS	PoS
Мова програмування смарт-контрактів	Rust, Golang, Solidity, TypeScript	Solidity, Vyper	Solidity
Швидкість транзакцій (TPS)	1000-12500 (залежить від обраного алгоритму консенсусу та налаштувань)	15-30 (очікується значне збільшення завдяки шардингу та іншим покращенням масштабування)	До 7000 (з можливістю збільшення за допомогою різних конфігурацій рішень Layer 2)
Масштабованість	Висока, забезпечується за допомогою IBC та модульної архітектури	Обмежена, вирішується за допомогою Layer 2	Висока, забезпечується за допомогою zk-Rollups

Вартість транзакцій	Відносно низькі, залежать від конкретної реалізації	Високі, залежать від навантаження мережі	Низькі, значно нижчі, ніж в Ethereum
Сумісність	Висока сумісність між блокчейнами через IBC	EVM-compatible; З іншими блокчейнами через мости	EVM-compatible; З іншими блокчейнами через мости
Розширюваність	Дуже висока, завдяки модульній архітектурі та IBC	Обмежена, потребує оновлень та Layer 2	Висока, з використанням різних Layer 2

Таблиця 1.1 — Порівняльна таблиця протоколів, в екосистемі яких можливе використання ZKP

Також варто зазначити, що у той час коли Ethereum та Polygon є повністю незалежними та монолітними мережами, то всю екосистему cosmos-based мереж можна вважати єдиним організмом з мережами, які заточені на певний ряд функцій (DeFi, DAO, NFT, Web3 Gaming та інші), що досягається за допомогою протокола IBC[12] (Inter-Blockchain Communication). Даний протокол додається як окремий модуль та дозволяє передавати будь-які дані між мережами (з цим модулем) та навіть виконувати транзакції та отримувати їх результати у мережі, з якої її відправили.

Рисунок 1.1 — Схема мереж екосистеми Cosmos з'єднаних протоколом IBC[8]

Основна частина

2.1 Постановка задачі

Інтеграція функціоналу Polygon ID для забезпечення можливості створення та зберігання доказів з нульовим розголошення з використанням імплементації zk-SNARKs на cosmos-based чейні полягає в створенні модуля для виконання необхідних смарт-контрактів та адаптації існуючих смарт-контрактів для їх деплою у створенні мережі та безпосередній деплой усіх необхідних контрактів.

Під час створення відповідного модуля треба буде враховувати синхронізацію виконання звичайних транзакцій, що виконуються стандартними модулями Cosmos SDK, та транзакцій, які виконуватимуться у середовищі EVM(Ethereum Virtual Machine). Також буде необхідність зберігання стану з середовища EVM після кожного створеного блоку та обробки помилок, які можуть бути повернені з EVM під час виконання відповідних транзакцій.

Для коректної роботи з новим модулем та його зворотної сумісності з іншими EVM-based мережами (EVM-compatible) треба буде імплементувати такий самий інтерфейс взаємодії, використовуючи стандартний EVM RPC для отримання даних (GET Requests) та відправлення транзакції.

Модуль має забезпечувати дані можливості:

- створення EVM середовища;
- EVM-compatible RPC protocol;
- можливість виклику транзакцій та запросів на отримання даних із середовища Cosmos SDK;
- State transition стану EVM;

- перерахунок витраченого газу у середовищі EVM у внутрішній газ системи Cosmos SDK;
- можливість встановлення спеціальних комісій на певні типи транзакцій у середовищі EVM.

Детальний опис алгоритмів виклику та зберігання стану EVM будуть описані у подальших розділах.

2.2 Опис алгоритмів

Дослідження існуючих мереж, побудованих за допомогою Cosmos SDK, які використовують різні імплементації EVM модуля, дали змогу сформувати деякі визначення та необхідні функції, що мають бути імплементовані у цьому модулі. Під час дослідження та формування вимог до модуля були досліджені такі блокчейн мережі:

- Evmos
- Haqq
- Fxcore
- Ethermint.

Нижче представлений список необхідних функцій та вимог до модуля з детальним описом:

- Віртуальна машина Ethereum[9] (EVM)

  Обчислювальний механізм, який можна розглядати як єдину сутність, на якому працює клієнт Ethereum. Як віртуальна машина (VM) EVM відповідає за детерміноване обчислення змін стану незалежно від середовища (апаратне забезпечення та ОС). Це означає, що кожен вузол має отримати абсолютно однаковий результат за однакового початкового стану та транзакції (tx).

  EVM вважається частиною протоколу Ethereum, яка керує розгортанням і виконанням смарт-контрактів. Однак EVM — це кінцевий автомат, який визначає правила для обчислення нового дійсного стану від блоку до блоку. Це ізольоване середовище виконання: означає, що код, який виконується всередині EVM, не має доступу до мережі, файлової системи чи інших процесів (зовнішніх API).

  Модуль повинен реалізовувати EVM як модуль Cosmos SDK. Це дозволить користувачам взаємодіяти з EVM, надсилаючи Ethereum

транзакції та виконуючи їх операції (messages) в заданому стані, щоб викликати перехід стану.

- Стан Ethereum

  Це структура даних, реалізована як Merkle Patricia Tree, яка зберігає всі облікові записи в ланцюжку. EVM вносить зміни в цю структуру даних, що призводить до нового стану з іншим коренем стану (state root). Тому Ethereum можна розглядати як ланцюжок станів, який переходить з одного стану в інший шляхом виконання транзакцій у блоці за допомогою EVM.

  Новий блок з транзакціями можна описати через його заголовок блоку (block header) (parent hash, block number, time stamp, nonce, receipts,...).

- Облікові записи[10]

  Є два типи облікових записів, які можуть зберігатися в стані за певною адресою:

  - Обліковий запис у зовнішній власності (EOA): має nonce (лічильник передачі) і баланс;
  - Смарт-контракт: має nonce, баланс, (незмінний) хеш коду, корінь зберігання (інша Merkle Patricia Trie).

  Смарт-контракти схожі на звичайні облікові записи в блокчейні, які додатково зберігають виконуваний код у двійковому форматі Ethereum, відомому як байт-код EVM. Зазвичай вони написані мовою високого рівня Ethereum, такими, як Solidity або Vyper, яка компілюється до байт-коду EVM і розгортається в блокчейні шляхом надсилання транзакції за допомогою клієнта Ethereum.

- Архітектура EVM

  EVM працює як стекова машина. Його архітектура складається з таких основних компонентів:

- віртуальна ППЗ: код контракту завантажується в цю пам’ять лише для читання під час обробки файлів txs;
- стан машини (волатильний): змінюється під час роботи EVM і очищається після обробки кожної транзакції;
- лічильник програми (program counter);
- газ: відстежує кількість використаного газу;
- стек і пам'ять: зміни стану обчислень;
- доступ до сховища облікового запису (постійний).

- Переходи між станами за допомогою смарт-контрактів

Зазвичай смарт-контракти надають загальнодоступний ABI, який є списком підтримуваних способів взаємодії користувача з контрактом.

Щоб взаємодіяти з контрактом і викликати перехід стану, користувач надсилає транзакцію з будь-якою кількістю газу та корисним навантаженням даних, відформатованих відповідно до ABI, із зазначенням типу взаємодії та будь-яких додаткових параметрів. Коли отримано транзакцію, EVM виконує байт-код EVM смарт-контрактів, використовуючи корисне навантаження транзакції (аргументи).

- Виконання байт-коду EVM

Байт-код EVM контракту складається з базових операцій (add, multiply, store тощо), які називаються кодами операцій (Opcodes). Кожне виконання коду операції вимагає газу, який потрібно оплатити вказавши відповідне значення у транзакції.

Таким чином, EVM вважається наближено повним за Тьюрингом, оскільки він допускає будь-які довільні обчислення, але кількість обчислень під час виконання контракту обмежена кількістю газу, наданого в транзакції. Вартість газу для кожного коду операції відображає вартість виконання цих операцій на

фактичному апаратному забезпеченні комп'ютера (наприклад, ADD = 3gas і SSTORE = 100gas).

Щоб розрахувати використання газу для транзакції, вартість газу множиться на ціну газу, яка може змінюватися залежно від попиту мережі на даний момент. Якщо мережа перебуває під великим навантаженням, можливо, вам доведеться заплатити вищу ціну за газ, щоб виконати транзакцію. Якщо досягнуто ліміту газу (out of gas exception), жодних змін до стану Ethereum не застосовується, за винятком того, що nonce відправника збільшується, а його баланс зменшується, щоб заплатити за марну трату часу EVM.

Смарт-контракти також можуть викликатися іншими смарт-контрактами. Кожен виклик нового контракту створює новий екземпляр EVM (включаючи новий стек і пам'ять). Кожен виклик передає стан пісочниці до наступного EVM. Якщо газ закінчується, усі зміни стану відхиляються. В іншому випадку вони зберігаються.

- StateDB

  Інтерфейс *StateDB* від go-ethereum представляє базу даних EVM для запиту усіх даних стану. Перехід стану EVM виконується за допомогою цього інтерфейсу, який у модулі має бути реалізованим у *Keeper*. Реалізація цього інтерфейсу зробить наш модуль EVM-сумісним.

- Transaction Logs

  У кожній транзакції результат містить логи Ethereum від виконання кінцевого автомата, які використовуються сервером JSON-RPC Web3 для запитів фільтрів і обробки хуків EVM. Логи транзакції зберігаються в тимчасовому сховищі під час виконання транзакції, а потім мають бути надіслані через івенти cosmos після обробки транзакції. Вони можуть бути запитані через gRPC і JSON-RPC.

- Block Bloom

  Block bloom - це структура даних, яка використовується в блокчейнах на основі Ethereum для швидкої перевірки того, чи певні транзакції або події відбулися у конкретному блоці. Вона базується на Блум-фільтрі (Bloom filter), який є ймовірнісною структурою даних, що дозволяє перевірити, чи міститься елемент у множині.

  Значення *Bloom* блоку зберігається в тимчасовому сховищі, а потім публікується (emit) через cosmos івент під час обробки *EndBlock*. При необхідності перевірити наявність певної події або логів, клієнт може швидко перевірити block bloom замість того, щоб проходити через всі транзакції блоку. Фільтри можуть бути запитані через gRPC і JSON-RPC.

Таким чином, враховуючи усі вищезазначені критерії та функції, що має містити модуль, можемо сформувати алгоритм виконання EVM-compatible транзакції у нашому модулі:

1. Користувач відправляє транзакцію, вказавши всі необхідні аргументи, кількість газу та підписавши транзакцію своїм приватним ключем.
2. Транзакція відправляється у мемпул (mempool) ноди та поширюється усім валідаторам мережі.
3. Пропоузер блоку обирає надіслану транзакцію для додавання її у поточний блок.
4. Додаток, побудований на базі Cosmos SDK, роутить виконання даної транзакції до нашого модуля EVM.
5. У функції CheckTx перевіряється валідність усіх даних та аргументів, переданих користувачем, та валідність підпису (відповідність адреси відправника публічному ключу, отриманому з підписа транзакції).

6. Message Server модуля EVM роутить виконання транзакції відповідному хедлеру (handler).
7. У хедлері створюється інстанс StateDB, який має доступ до стореджа з повним станом системи, та інстанс EVM, якому передається StateDB та виділяється пам’ять для тимчасового зберігання проміжних результатів транзакції, логів та кінцевих результатів виконання транзакції.
8. Виклик EVM з надісланою користувачем транзакцією.
9. Збереження результатів виконання транзакції та збереження нового стану StateDB, якщо транзакція була успішна.
10. Повернення користувачу коштів за газ, що не був використаним під час виконання транзакції.
11. Збереження та публікація Block Bloom івентів у EndBlocker.

2.3 Огляд взаємодії з Polygon ID

Інфраструктура Polygon ID складається з Емітента (Issuer), Верифікатора (Verifier) та блокчейну, на якому розгорнуті необхідні контракти для зберігання поточного стану Issuer, що використовується для перевірки Verifier-ом валідності клеймів (тверджень) користувача.

Емітент[11] - це будь-який суб’єкт, який видає підтверджувані облікові дані (Verifiable Credentials). Ви можете розглядати облікові дані як заяву: те, що емітент говорить про інший суб’єкт. Наприклад, коли університет (емітент) стверджує, що студент (суб’єкт) має ступінь, це verifiable credential.

Емітентом може бути:

- DAO (децентралізована автономна організація), що видає своїм членам «заяви про членство»;
- державна установа, яка видає своїм громадянам документи, що посвідчують особу;
- застосунок машинного навчання для розпізнавання обличчя, який видає «докази особи»;
- Роботодавець, який підтверджує своїх працівників.

Верифікатор - це будь-яка Web2 або Web3 платформа, яка може аутентифікувати користувачів на основі їхніх облікових даних.

Верифікатори можуть налаштовувати запити на основі існуючих облікових даних користувачів, зібраних від широкого кола емітентів. Запит охоплює критерії, яким користувач повинен відповідати для аутентифікації, наприклад, "повинен бути членом певного DAO" або "повинен бути старшим за 18 років". Polygon ID забезпечує

безперебійний, індивідуальний і орієнтований на конфіденційність процес аутентифікації для користувачів.

Запит верифікатора створюється за допомогою повноцінної мови запитів zk[13] (zk Query Language) та інкапсулюється в QR-код (або через глибоке посилання) для зчитування користувачем. Користувач сканує QR-код за допомогою свого гаманця, щоб ініціювати генерацію доказу.

Процес верифікації не передбачає жодної взаємодії між верифікатором і емітентом запитуваних облікових даних. У рамках запиту верифікатор включає ідентифікатори довірених емітентів. Наприклад, верифікатор повинен додати ХНУ імені В. Н. Каразіна як єдиного довіреного емітента під час перевірки, чи є особа студентом. Університету не потрібно приймати чи взаємодіяти з верифікатором.

Наприкінці процесу верифікатор отримує криптографічний доказ того, що користувач відповідає запиту, при цьому користувач ділиться лише мінімально можливою кількістю даних, необхідних для взаємодії.

Також користувачу буде необхідний цифровий гаманець, з функціями створення акаунта користувача, безпечного зберігання всіх приватних даних користувача, зберігання усіх Verifiable credentials, створених відповідними Емітентами та створення доказів з нульовим розголошенням для передачі їх Верифікаторам та інше.

Процес взаємодії даних компонентів буде виглядати наступним чином:

1. Користувач аутентифікується у Емітента та запитує дані або твердження, які необхідно буде додати у Verifiable credential (VC).
2. Емітент випускає необхідний Verifiable credential та публікує його публічну інформацію, таку як ID, DID (Decentralized Identifier)

користувача, дату видачі та дату до якої VC дійсний та інше, у смарт-контракті на блокчейні.

3. Користувач зберігає у себе VC та при необхідності верифікації створює доказ з нульовим розголошенням, використовуючи відповідний VC.

4. Верифікатор отримує доказ від користувача, перевіряє його, користуючись публічними даними, отриманими зі смарт-контракту (факт випуску VC, дати випуску та дійсний до, те що VC не був відкликаний Емітентом) та надає доступ користувачу до певних послуг.

Верифікатор жодним чином не взаємодіє з Емітентом у даному флоу, але Верифікатор “довіряє” лише певному списку Емітентів (наприклад, Верифікатор, який “довіряє” тільки українським університетам, не прийме VC від італійського університету).

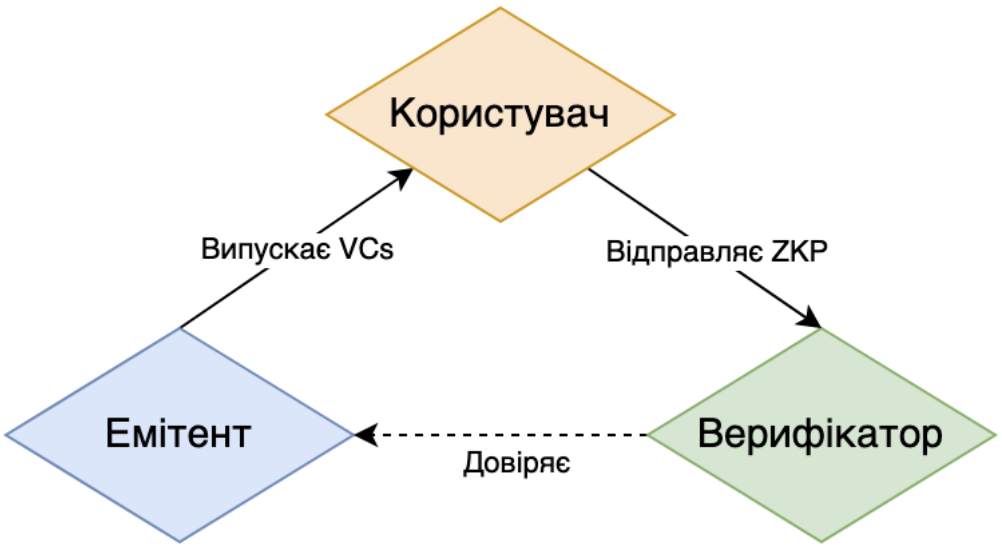

Схема 2.1 — Схема взаємодії компонентів протоколу Polygon ID

2.4 Деплой контрактів та тестування

Наступними кроками дослідження доказів з нульовим розголошенням у cosmos-based мережах є:

1. Додавання створеного EVM модуля до стандартного додатку, що побудований на базі Cosmos SDK.
2. Запуск локальної мережі та перевірка того, що валідатори мережі здатні досягати консенсусу та створювати нові блоки та перевірка стандартних транзакцій, таких, як відправка токенів, делегування токенів обраним валідаторам тощо.
3. Деплой контрактів необхідних для функціонування інфраструктури Polygon ID.
4. Порівняння використання газу у Polygon та нашій cosmos-based мережі.
5. Створення Verifiable credentials з використанням стандартного Емітента.
6. Верифікація Zero Knowledge proof-ів Верифікатором.

Після додавання створеного модуля EVM та налаштування трьох валідаторів для підтримки консенсусу та створення блоків була запущена локальна мережа. Посилання на код ноди валідатора мережі подано у Додаток А.

Рисунок 2.1 — Логи ноди валідатора мережі під час створення блоків

Із рисунку 2.1 вище можемо зробити висновок, що мережа із доданим модулем EVM успішно запустилася та 3 валідатори досягли консенсусу та почали створювати блоки.

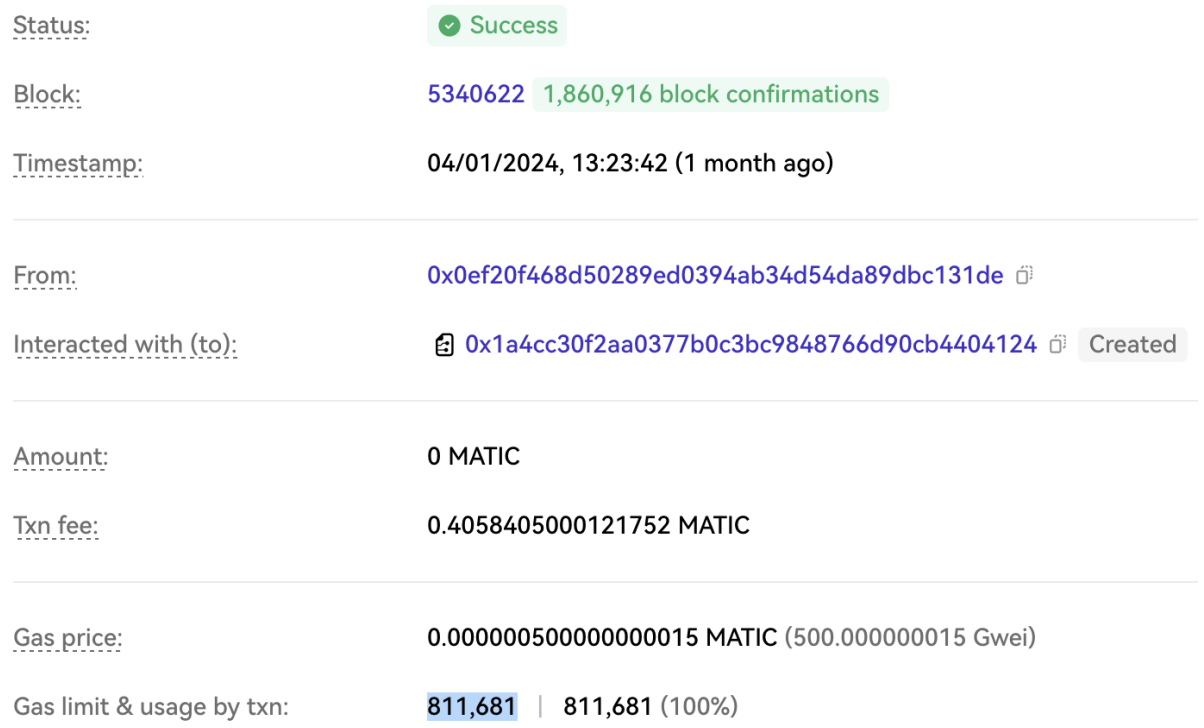

Рисунок 2.2 — Результати транзакції розгортання смарт-контракту стану Емітента у мережі Polygon

```
- attributes:
  - key: sender
    value: evm17xpfvakm2amg962yls6f84z3kell8c5ljcjw34
  type: message
- attributes:
  - key: amount
    value: 0
  - key: ethereumTxHash
    value: 0x2e2e19a1a22ccf419e7c350f0c3476e27b79b78e12bddd0b15fca58caab35454
  - key: txIndex
    value: 0
  - key: txGasUsed
    value: 811681
  - key: txHash
    value: 5ECE66DC56B8A96F499817F4876BCECC58A94AFA7E96FD1F150B5545371FA263
  - key: recipient
    value: 0xc5bA89c3ebb1A909604386Be24941309Bd4df8D1
```

Рисунок 2.3 — Результати транзакції розгортання смарт-контракту стану Емітента у cosmos-based мережі

На скрінах вище показано результати розгортання контракту стану Емітента у мережі (Issuer on-chain state contract). Рисунок 2.2 - результати розгортання контракту у мережі Polygon, рисунок 2.3 - у локальній cosmos-based мережі зі створеним модулем EVM.

Також із результатів транзакції можна помітити, що на деплой одного і того самого контракту у різних мережах була витрачена однакова кількість газу, що вказує на те, що EVM модуль працює коректно, оскільки у обох мережах використовується однакова версія EVM та дана поведінка передбачувана.

Наступним кроком до локального Емітента було надіслано запит на створення Verifiable credential. У результаті цього запиту був створений VC, збережені його дані у базі даних Емітента та опублікований стан Емітента у створену мережу.

 ce842205-9a83-11ee-bd91-0242c0a83006 | did:polygonid:polygon:amoy:2qQPv3iiEqqjxiMQ3q2Sqv512rsYLK3ZSy9bdW9eHi | did:polygonid:polygon:amoy:2q
QPv3iiEqqjxiMQ3q2Sqv512rsYLK3ZSy9bdW9eHi | c9b2370371b7fa8b3dab2a5ba81b6838 | 1702555032 | f | 3521314513 | ["14366836738280263696847 39
6792315741188809","22993833389585891901791499727402251184400569823566302749372448127312335362","182940450241757831197993683460604752468962013
573739891707457801456484545613 3","0","31406596946710376013805846225","0","0","0"] | {"mtp": {"siblings": ["1076888434368811207929930451841065
4477374339923415955096855402550312475124907"], "existence": true}, "type": "Iden3SparseMerkleTreeProof", "coreClaim": "c9b2370371b7fa8b3dab2a
5ba81b68382a00000000000000000000000000000002120aa41eb0120558af9c95cd8c5b27cbef0ac0d13f6f29aa7eafae98030d00058ba5f87b4ac962aa950195f0a470ebaa6
7c7b9a31c657ffaa61a5300680b040000000000000000000000000000000000000000000000000000000000000000d1fee2d10000000098ed7a65000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000", "issuerData": {"id": "did:polygonid:polygon:amoy:2qQPv3iiEqqjxiMQ3q2Sqv
512rsYLK3ZSy9bdW9eHi", "state": {"txId": "0x443026c06c6022d0b6dbd1f08c18e2713c06f76fb9a7a84172a5afb5bba71409", "value": "0254fc7b227b00eef8ce
abc745025bae51eae8a3d30807dc0594719d02b3c50a", "blockNumber": 32897, "rootOfRoots": "39ea24ed8b616da568bee626db64d83182aa024e100edc679279b04e
85152d1a", "blockTimestamp": 1716052008, "claimsTreeRoot": "e81cf92f4c56ec5213ce51c6cdafe670fb869f5f4f5d837e7c1b42817a67ca0f", "revocationTre
eRoot": "0000000000000000000000000000000000000000000000000000000000000000"}}} | {"type": "BJJSignature2021", "coreClaim": "c9b2370371b7fa8b3d
ab2a5ba81b68382a00000000000000000000000000000002120aa41eb0120558af9c95cd8c5b27cbef0ac0d13f6f29aa7eafae98030d00058ba5f87b4ac962aa950195f0a470e
baa67c7b9a31c657ffaa61a5300680b040000000000000000000000000000000000000000000000000000000000000000d1fee2d10000000098ed7a6500000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000", "signature": "529ea2edffae30e99e4431ac40ef8f638502c7a0141d1d911f398
61c9ce17a8bf239cf82e1a48503e90fee69f11de0910ddf84eeef130b3a9dafed7d710cc904", "issuerData": {"id": "did:polygonid:polygon:amoy:2qQPv3iiEqqjxi
MQ3q2Sqv512rsYLK3ZSy9bdW9eHi", "mtp": {"siblings": [], "existence": true}, "state": {"value": "71ed9d8bb9a0b61a5ab8363b7c924190dec158e6c21aa2
189d803f023a6eeb1a", "claimsTreeRoot": "104388b9012340a1fa2cfe36ab1705cdda7735e8a9924392b46208546908de04"}, "authCoreClaim": "cca3371a6cb1b71
5004407e325bd993c000000000000000000000000000000000000000000000000000000000000000000000000000000000000ec617d2c77153ba265183f021e2d
903f58663ab44fecc367c9dc627e430bf921dff3736f49a5f963090bfd2b2453d053350dbb523d25c8709e7160b74270e62900000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000", "credentialStatus": {"id": "http://localhost:3001/v1/did%3Apolyg
onid%3Apolygon%3Amumbai%3A2qKmEFFHJdbP3e6kCPo78TCNVrz51kBrwimTpwe1go/claims/revocation/status/0", "type": "SparseMerkleTreeProof", "revocatio
nNonce": 0}}} | | 0254fc7b227b00eef8ceabc745025bae51eae8a3d30807dc0594719d02b3c50a | {"id": "http://localhost:3001/v1/did%3Apolyg
onid%3Apolygon%3Amumbai%3A2qKmEFFHJdbP3e6kCPo78TCNVrz51kBrwimTpwe1go/claims/revocation/status/3521314513", "type": "SparseMerkleTreeProof", "
revocationNonce": 3521314513} | 113708739863514385176503964053068854711393622698042632367189859996645883308 82 | t

Рисунок 2.4 — Дані VC з бази даних Емітента

На рисунку 2.4 представлені поля: id, identifier, issuer, schema_hash, expiration, revoked, rev_nonce, core_claim, mtp_proof, signature_proof, merkle_root, identity_state, credential_status, index_hash, mtp.

Поле, яке має найважливіше значення, mtp_proof заповнюється тільки після публікації стану Емітента до мережі та містить інформацію

про Merkle path (набір хешів, який забезпечує шлях від хешу виданого VC до Merkle root), що дає змогу перевірити те, що Verifiable credential був випущений.

Далі додаємо наш Verifiable credential до додатку Polygon ID з налаштованою нашою мережею

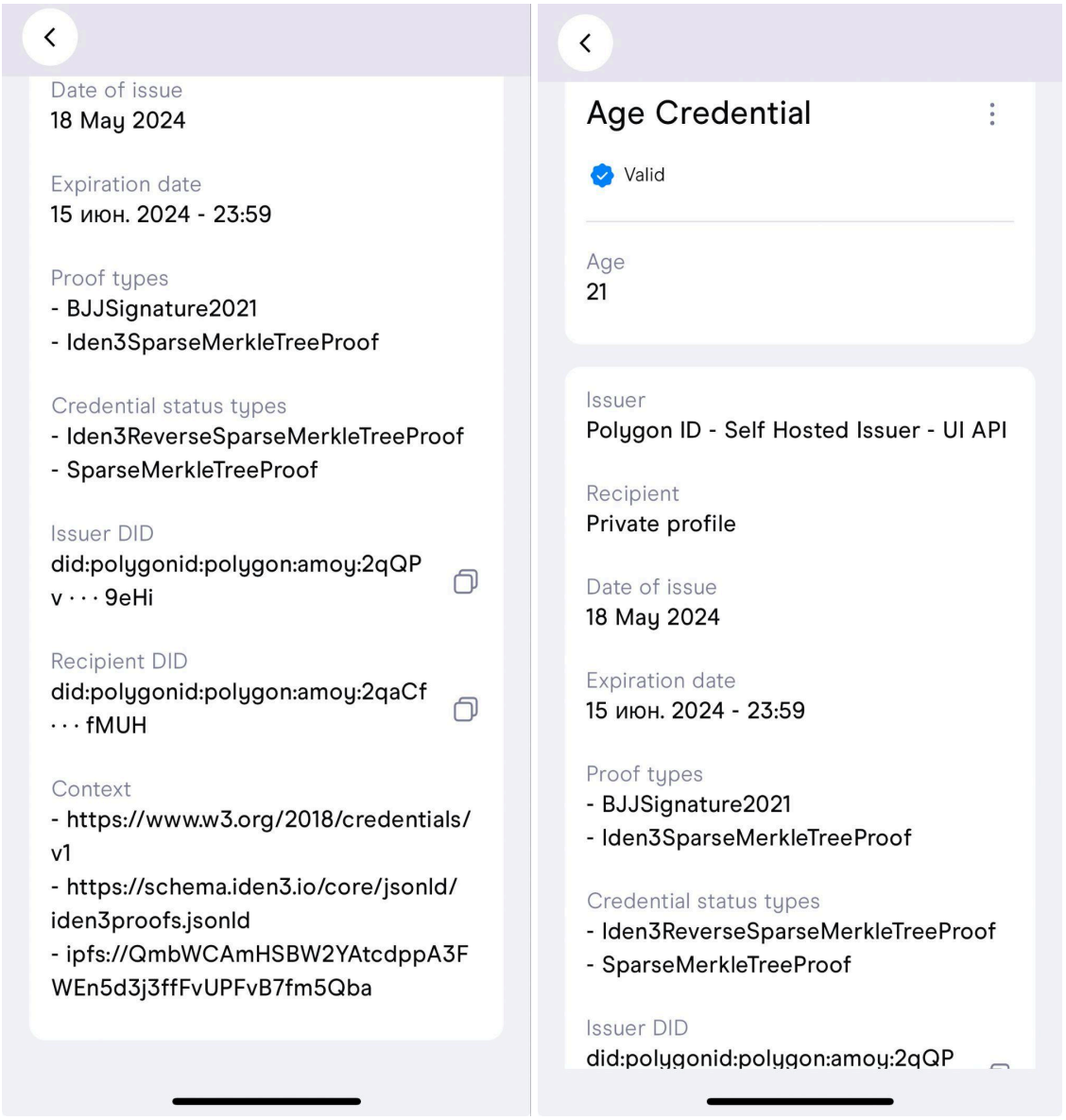

Рисунок 2.5 — Verifiable credential доданий до додатку Polygon ID

Створений VC був успішно доданий до додатку Polygon ID, що свідчить про те, що він був створений коректно та пройшов верифікацію даного додатку (при спробі додавання невалідного Verifiable credential або який прострочений, додаток поверне помилку). З даних про Verifiable

credential можемо побачити, що він містить Iden3SparseMerkleTreeProof, це повідомляє, що даний Verifiable credential був доданий до дерева Меркла стану Емітента у створеній cosmos-based мережі.

Тестування розробленого модуля можна вважати успішним, для подальшого використання та розвитку ZKP у cosmos-based мережах необхідне створення Верифікаторів для створений доказів з нульовим розголошенням та виділення доменів для деплою Емітентів для публічного користування протоколом Polygon ID.

2.5 Перспективи подальшого розвитку ZKP у cosmos-based мережах

Створення власного модуля EVM у cosmos-based мережі з подальшою інтеграцією Polygon ID продемонструвало здатність використання ZKP на базі власної cosmos-based мережі.

Дане рішення не є ідеальним та найефективнішим способом інтеграції доказів з нульовим розголошенням у мережах на базі Cosmos SDK, бо інтегрує у себе EVM, що не є “природним” компонентом для cosmos-based мереж та додає багато зайвого функціоналу, якщо розробники мережі бажають отримати лише можливість користуватися усіми перевагами доказів з нульовим розголошенням.

Більш оптимальним рішенням буде розробка власного модуля ZKP, який підтримуватиме лише необхідні функції та повністю відповідатиме архітектурі Cosmos SDK. Також дане рішення дозволить модулю мати прямий доступ до сховища даних з поточним станом системи та не виділяти багато додаткової тимчасової пам’яті для проміжного стану системи та вивантаження у нього додаткових смарт-контрактів, як у випадку використання EVM. Дане рішення зробить розробку нових функцій та тестування продукту прозорішим, з можливістю зручного логування та дебагінгу, виключивши з процесу виконання транзакцій етап з чорною скринькою у вигляді EVM, що зробить розробку ZKP рішень на базі cosmos-based мереж зручнішою, ніж на базі EVM-compatible мереж, у яких відсутня можливість зручного дебагінгу.

Розробка модуля вузькоспеціалізованого на ZKP дозволятиме гнучко налаштовувати комісії за певні типи транзакцій, наприклад дозволити певному списку Емітентів відправляти транзакції оновлення стану без комісій, з можливістю змінювати цей список, використовуючи голосування у мережі з пропорційним правом голосу усіх користувачів, що застейкали свої токени у мережі.

Реалізація интерфейсу протоколу IBC у даному модулі дасть змогу мережам, що встановлять зв’язок зі створеною мережею, використовуючи протокол IBC, отримувати поточний стан Емітентів для забезпечення on-chain верифікації доказів з нульовим розголошенням користувачів без необхідності додавати до своєї мережі ZKP модуль, а також виступати у ролі on-chain Емітентів та зберігати свій стан у мережі із ZKP модулем, що забезпечує можливість верифікації ZKP, що стосуються даної мережі, у інших мережах екосистеми Cosmos.

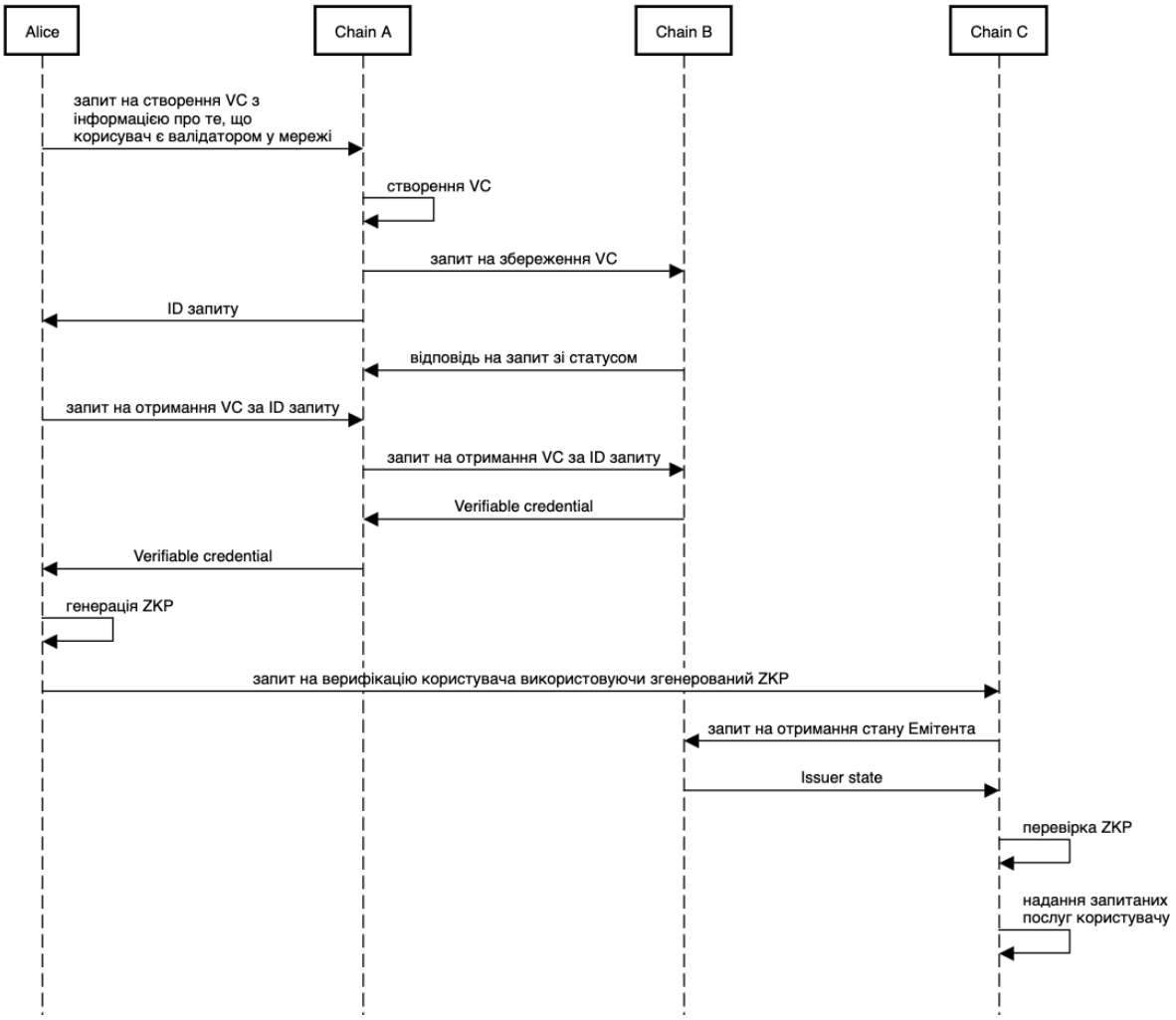

Схема 2.2 — Діаграма використання доказів з нульовим розголошенням з протоколом IBC

Наприклад, Chain B - мережа із ZKP модулем, Chain A - Емітент (Issuer), Chain C - Верифікатор (Verifier). За допомогою модуля ZKP та

з’єднання мереж використовуючи протокол IBC можливо створення твердження про користувача у мережі A з подальшим використанням даного твердження у мережі C, у даної концепції є декілька переваг:

- відсутність необхідності зберігання інформації про мережу A у мережі C, такої як chain-id, кодек для конвертування bech32 адрес різних мереж, IBC з’єднання з мережею A тощо;
- відсутність необхідності надання додаткової інформації про користувача;
- абстрактність протоколу (можливість розширення списку підтримуваних мереж без необхідності оновлення мережі);
- уніфікований спосіб верифікації певних даних користувача (ролі користувача у інших мережах, його платоспроможність, активність у екосистемі Cosmos тощо).

Дана концепція дає можливість створювати Digital Identity користувача, що є однією з найактуальніших проблем у блокчейні. Digital identity користувача може бути використана для врегулювання дій користувача у мережах блокчейну, що може сприяти прийняттю та інтеграції технології блокчейну у повсякденне життя на законодавчому рівні, бо саме необмежена та безкарна анонімність користувачів у мережах на основі технології блокчейну відштовхує владу різних країн від сприяння розвитку та інтеграції у суспільство даних технологій. Також digital identity може бути використана для програм лояльності для активних користувачів, для перевірки платоспроможності користувачів, для організації децентралізованих голосувань певної групи користувачів та багато іншого.

Також є багато інших застосувань ZKP у cosmos-based мережах:

- zk-Rollups

  У екосистемі Cosmos використовується горизонтальне масштабування за допомогою протоколу IBC, завдяки zk-Rollups

буде доступне вертикальне масштабування, що підвищить пропускну здатність окремо взятої мережі.

- Bulletproofs

  За допомогою даної технології буде можливе створення мережі на базі Cosmos SDK із анонімними балансами користувачів, з підтримкою перевірки коректності транзакцій переміщення токенів, але без розкриття їх сум.

- Web3 Gaming

  Можливості зберігання результатів та перевірки їх без розкриття деталей гри, що сприяє підвищенню чесності ігор та довіри до них.

- Децентралізовані фінансові послуги (DeFi)

  Дозволяють перевіряти стан активів і зобов’язань без розкриття деталей, що особливо важливо для DeFi-платформ, які надають кредити та позики.

- Комплаєнс

  ZKP можуть використовуватися для підтвердження відповідності вимогам (наприклад, KYC/AML) без розкриття особистих даних користувачів.

- Захист комерційних таємниць

  ZKP нададуть можливість виконувати умови контрактів без розкриття комерційних таємниць та конфіденційної інформації.

Завдяки протоколу IBC не буде необхідності імплементувати власний модуль ZKP кожній мережі, а буде можливість використовувати одну або декілька мереж з даним модулем як хаби інформації та довіри.

Висновки

В ході кваліфікаційної роботи були досліджені cosmos-based мережі, докази з нульовим розголошенням та Polygon ID, як фреймворк для швидкого розгортання підтримки доказів з нульовим розголошенням у мережі. Був розроблений EVM модуль для мереж, побудованих на базі Cosmos SDK, який надав підтримку смарт-контрактів, зробив мережу EVM-compatible та надав можливості для розгортання протоколу Polygon ID у cosmos-based мережі.

Була запущена локальна cosmos-based мережа зі створеним EVM модулем, на базі якої був розгорнутий протокол Polygon ID та проведено тестування.

У ході тестування були перевірені можливості валідаторів досягати консенсусу та створювати нові блоки у мережі з новим модулем, були розгорнуті контракти для роботи з інфраструктурою Polygon ID, створені Verifiable credentials за допомогою стандартного Емітента Polygon ID. Під час розгортання смарт-контрактів та взаємодії з ними були проведені порівняння витрати газу на дані транзакції з таким самим контрактом, розгорнутим у мережі Polygon для перевірки коректної роботи створеного модуля EVM. Під час тестування витрати газу у cosmos-based мережі та у мережі Polygon були однакові, що свідчить про коректну роботу створеного модуля та відсутність марної витрати газу.

Оскільки у сучасному світі питання конфіденційності та безпеки даних стають ще важливішими, технологія Zero-knowledge proof-ів стає дедалі важливішою та знаходить ще більше застосувань у повсякденному житті. Інтеграція даної технології у екосистему Cosmos надає безліч нових можливостей для використання її як on-chain, так і в реальному світі. Саме завдяки архітектурі cosmos-based мереж та екосистеми в цілому, інтеграція доказів з нульовим розголошенням є перспективним

шляхом розвитку даної екосистеми та надає більше можливостей, ніж у EVM-compatible мережах, таких як Ethereum або Polygon.

У даній роботі представлені перспективи подальшого розвитку ZKP у cosmos-based мережах та реальні приклади застосування даної технології з урахуванням особливостей та переваг cosmos-based мереж.

Дана робота є першим кроком до повної інтеграції технології доказів з нульовим розголошенням у екосистему Cosmos. Вона може бути використана як основа для подальшого розвитку технології ZKP як у теоретичному плані, так і у практичному.

Список використаних джерел

1. Polygon ID documentation. Overview: веб-сайт. URL: https://devs.polygonid.com/docs/introduction/ (дата звернення: 10.04.2024)
2. StarkNet documentation: веб-сайт. URL: https://devs.polygonid.com/docs/introduction/ (дата звернення: 11.05.2024)
3. zkSync documentation. Concepts overview: веб-сайт. URL: https://docs.zksync.io/zk-stack/concepts/overview.html (дата звернення: 11.05.2024)
4. What is Monero? Monero: веб-сайт. URL: https://www.getmonero.org/get-started/what-is-monero/ (дата звернення: 11.05.2024)
5. Cosmos SDK documentation. High-level overview: веб-сайт. URL: https://docs.cosmos.network/main/learn/intro/overview (дата звернення: 05.04.2024)
6. Tendermint Core documentation. Overview: веб-сайт. URL: https://docs.tendermint.com/v0.34/tendermint-core/ (дата звернення: 05.04.2024)
7. What is CometBFT? CometBFT: веб-сайт. URL: https://docs.cometbft.com/v0.38/introduction/ (дата звернення: 09.04.2024)
8. Map of zones. Map of zones: веб-сайт. URL: https://mapofzones.com/home?columnKey=ibcVolume&period=24h (дата звернення: 15.05.2024)
9. Ethereum documentation. Ethereum virtual machine (EVM): веб-сайт. URL: https://ethereum.org/en/developers/docs/evm/ (дата звернення: 24.04.2024)

10. Ethereum documentation. Ethereum accounts: веб-сайт. URL: https://ethereum.org/en/developers/docs/accounts/ (дата звернення: 24.04.2024)
11. Polygon ID documentation. Issuer Overview: веб-сайт. URL: https://devs.polygonid.com/docs/issuer/issuer-overview (дата звернення: 10.05.2024)
12. IBC protocol documentation. Overview: веб-сайт. URL: https://ibc.cosmos.network/main/ibc/overview (дата звернення: 28.04.2024)
13. Polygon ID documentation. ZK Query Language: веб-сайт. URL: https://devs.polygonid.com/docs/verifier/verification-library/zk-query-language/ (дата звернення: 13.05.2024)

Додаток А

Код ноди cosmos-based мережі зі створеним модулем EVM доступний за посиланням: https://github.com/slandymani/evm-module