RaiBlocks: Криптовалютна Мережа Без Комісій

Colin LeMahieu
clemahieu@gmail.com

Анотація—В останній час при високому попиті і обмеженої маштабованості збільшився середній час транзакцій та комісій в популярних криптовалютах, що призвело до незадовільного досвіду. Тут ми познайомимо з RaiBlocks, криптовалюта з новою архітектурою блокової структури, де кожен аккаунт має свою власну блок-ланцюжок, забезпечуючи майже миттєву швидкість транзакції і необмежену масштабованість. У кожного користувача є своя блок-ланцюжок, дозволяючи їм асинхронно оновлюватися його для іншої мережі, що призводить до швидких транзакціях з мінімальними витратами. Транзакції відстежують залишок на акаунтах, а не суми в транзакції, дозволяючи агресивну обрізку бази даних без шкоди для безпеки. На сьогоднішній день мережа RaiBlocks опрацювала понад 4,2 млн транзакцій маючи повний обсяг бази трохи більше 1,7 ГБ. Безкомісійність RaiBlocks та транзакції в долі секунд роблять його головною криптовалютою для споживчих транзакцій.

Індексний термін—криптовалюта, блокчейн, raiblocks, розподілена база, цифровий, транзакції

I. ВСТУП

З МОМЕНТУ появи Bitcoin в 2009 році спостерігається зростаючий відхід від традиційних, підтримуваних урядом валют і фінансових систем, до сучасних систем платежів, заснованих на криптографії, які пропонують можливість зберігати і переводити кошти надійним та безпечним шляхом [1]. Для ефективного функціонування, валюта повинна легко передаватися, без змоги відміни і мати обмежені або зовсім без комісії. Збільшений час транзакцій, великі комісії і сумнівна масштабованість мережі викликали питання про практичність Bitcoin як повсякденної валюти.

У цій статті ми знайомимо з RaiBlocks - криптовалюту з малою затримкою, заснованої на інноваційній структурі даних блок-ланцюга, що пропонує необмежену масштабованість і відсутність транзакційних комісій. RaiBlocks це простий протокол з єдиною метою бути високоефективною криптовалютою. Протокол RaiBlocks може працювати на малопотужному обладнанні, дозволяючи бути практичною і децентралізованою криптовалютою для повсякденного використання.

Статистика криптовалюти, представлена в цьому документі, є точною на момент публікації.

II. ДОВІДКОВА ІНФОРМАЦІЯ

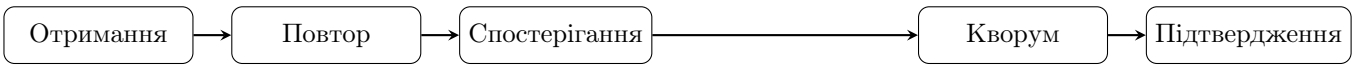
У 2008 році анонім під псевдонимом Satoshi Nakamoto опублікував білий папір, що описує першу в світі децентралізовану криптовалюту Bitcoin [1]. Ключовим нововведенням Bitcoin став блокчейн, публічна, незмінна і децентралізована структура даних, яка використовується в якості регістру операцій з валютою. На жаль, у міру того як Bitcoin розвивався, деякі проблеми в протоколі зробили Bitcoin недоступним для багатьох додаткі:

1) Погана масштабованість: Кожен блок блокчейну може зберігати обмежену кількість даних, що означає, що система може обробляти тільки стільки транзакцій в секунду, скільки вистачає місця в блоці для запису транзакцій. В даний час середній збір за транзакцію становить $10,38 [2].
2) Висока затримка: середній час підтвердження становить 164 хвилини [3].
3) Енерго неефективна: Мережа Bitcoin споживає близько 27.28 млрд. кВт/г на рік, використовує в середньому 260KWh на транзакцію [4].

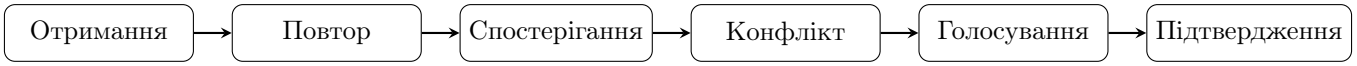
Bitcoin та інші криптовалюти функціонують шляхом досягнення консенсусу щодо свого глобального блокчейна, з тим щоб перевірити законність операції при протистоянні зловмисним. Bitcoin досягає консенсусу за допомогою економічному заходу, званий доказом роботи (PoW). В системі PoW учасники конкурують за обчислення числа, званого nonce, так що хеш всього блоку знаходиться в цільовому діапазоні. Цей допустимий діапазон обернено пропорційний сукупній обчислювальній потужності всієї мережі Bitcoin для підтримки узгодженого середнього часу, що витрачається на пошук допустимого значення nonce. Тоді знайшовши вірний номер nonce може додати блок в блокчейн; тому ті, хто володіє величезним обчислювальним ресурсом для обчислення nonce, відіграють велику роль в стані блокчейна. PoW забезпечує стійкість до атаки Sybil, де атакуючий поводиться як кілька об'єктів, щоб отримати додаткову потужність в децентралізованій системі, знижує рівень стану гонки, яка за своєю природою існує при доступі до глобальної структури даних.

Альтернативний консенсусний протокол це підтвердження частки участі (PoS), вперше був введений Peercoin в 2012 році [5].В системі PoS учасники голосують з вагою, еквівалентною кількістю коштів, які вони мають в даній криптовалюті. За допомогою цього механізму тому, хто має великі фінансові інвестиції, отримують більше впливу і за своєю суттю стимулюються підтримувати чесність системи або ризик втратити свої інвестиції. PoS позбавляє від марнотратної конкуренції потужності обчислень, вимагаючи тільки легкого програмного забезпечення, що працює на малій потужності.

Оригінальний документ RaiBlocks і перша бетаверсія були опубліковані в грудні 2014 року, що робить її одну з перших криптовалют використовую-



(а) Якщо конфлікт не виявлено, то ніяких додаткових дій не вимагається.



(б) У разі конфлікту транзакції ноди голосують за допустиму транзакцію.

Рис. 1. RaiBlocks не вимагає додаткових витрат для типових транзакцій. У разі конфлікту транзакцій ноди повинні голосувати за допустимі транзакції

чих Directed Acyclic Graph (DAG) [6]. Незабаром почали розвиватися інші криптовалюти DAG, зокрема DagCoin/Byteball і IOTA [7], [8]. Ці криптовалюти на основі DAG зламали форму блокчейна, покращивши продуктивність системи та безпека. Byteball досягають консенсусу, спираючись на «головний ланцюжок» складаються з чесних, авторитетних та довірених "користувачів"свідків, в той час як IOTA досягає консенсусу щодо PoW складеним транзакціям. RaiBlocks досягає консенсусу за допомогою зваженого голосування по конфліктуючих транзакціях. Ця система консенсусу забезпечує більш швидкі і більш детерміновані транзакції, зберігаючи при цьому сильну децентралізовану систему. RaiBlocks продовжує цю розробку і позиціонує себе як одну з найбільш високопродуктивних криптовалют.

III. КОМПОНЕНТИ RAIBLOCKS

Перш ніж описати загальну архітектуру RaiBlocks, ми визначимо окремі компоненти, які складають систему.

А. Аккаунт

Аккаунт - це частина публічного ключа від пари ключів цифрового підпису. Публічний ключ, також іменований адресою, передається іншим учасникам мережі, в той час як закритий ключ зберігається в секреті. Пакет даних з цифровим підписом гарантує, що вміст було дозволено відповідним власником закритого ключа. Один користувач може управляти багатьма аккаунтами, але для кожного аккаунта може існувати тільки один публічний адресу.

Б. Блок/Транзакції

Термін «блок» і «транзакція» часто використовуються як взаємозамінні, коли блок містить одну транзакцію. Транзакція конкретно відноситься до дії, тоді як блок відноситься до цифрового кодування транзакції. Транзакції підписуються закритим ключем, який належить аккаунту, на якому виконується транзакція.

В. Ledger

Ledger - це глобальний набір аккаунтів, де кожен аккаунт має свій власний ланцюжок транзакцій (рис 2). Це ключовий компонент дизайну, який підпадає під категорію заміни угоди про час виконання з угодою часу розробки; кожен погоджується з допомогою перевірки підпису, що тільки власник облікового запису може змінити свій власний ланцюжок. Це перетворює, судячи з вигляду загальну структуру даних, розподілений ledger, до набору не загальних (приватних) структур користування.

Г. Нода

Нода являє собою частину програмного забезпечення, що працює на комп'ютері, який відповідає протоколу RaiBlocks і бере участь в мережі RaiBlocks. Програмне забезпечення управляє ledger і будь-якими обліковими записами, яким може управляти нода, якщо такі є. Нода може або зберігати все ledger, або її обрізану історію, яка містить тільки останні кілька блоків ланцюжка кожного аккаунта. Під час налаштування нової Ноди рекомендується перевіряти всю історію і робити зріз локально.

IV. ОГЛЯД СИСТЕМИ

На відміну від блокчейнів які використовуються в багатьох інших криптовалютах, RaiBlocks використовує структуру блок-решітки. Кожен акаунт має свій власний блокчейн (Аккаунт- ланцюжок), еквівалент історії транзакцій/балансу аккаунта (Рис. 2). Кожен ланцюжок аккаунта може бути оновлений тільки власником

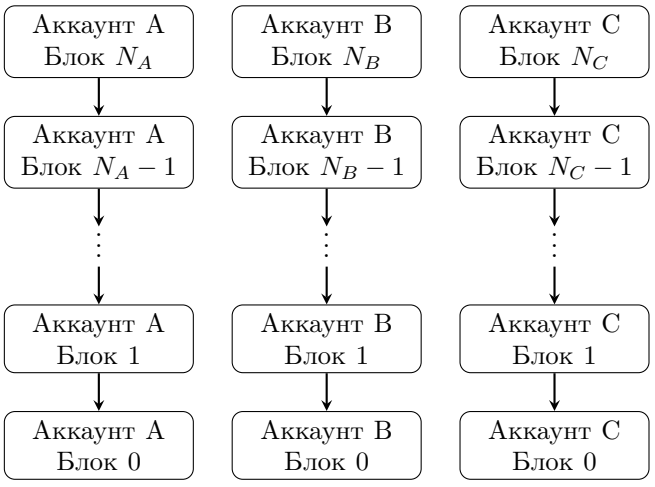


Рис. 2. Кожен акаунт має свій власний блокчейн, що містить історію зміни балансу. Блок 0 повинен бути відкритою транзакцією. (Секц IV-Б)

облікового запису; це дозволяє кожному ланцюжку аккаунта бути оновленим негайно і асинхронно до решти блок-решітці, що призводить до швидких транзакціях. Протокол RaiBlocks є надзвичайно легким; кожна операція вписується в необхідний мінімальний розмір пакета udp для передачі через інтернет. Вимоги до обладнання для нод також мінімальні, так як Ноди повинні тільки записувати і ретранслювати блоки для більшості транзакцій (Рис 1).

Система ініціюється з аккаунтом генезису, що містить баланс генезису. Баланс генезису є фіксованою кількістю і ніколи не може бути збільшений. Баланс генезису ділиться і відправляється на інші акаунти через транзакції відправки зареєстровані в ланцюжку аккаунта генезису. Сума залишків на всіх рахунках ніколи не перевищить початковий баланс генезису, що дає системі верхню межу по кількості і не дозволяє збільшити його.

У цьому розділі описується створення і поширення різних типів транзакцій в мережі.

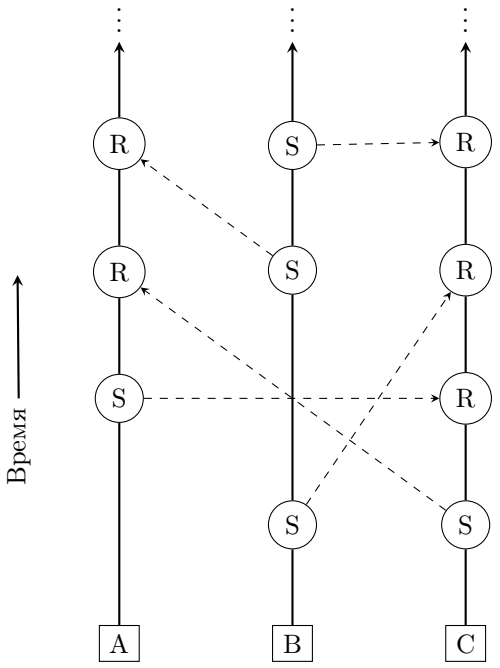


Рис. 3. Візуалізація блок-решітки. Кожен переказ коштів вимагає блоку відправки (S) і блоку отримання (R), кожен підписаний власником аккаунта-ланцюжка (A,B,C)

А. Транзакції

Переказ коштів з одного аккаунта на інший вимагає двох операцій: відправки списання коштів з балансу відправника і прийому додавання коштів на акаунт одержувача (рис. 3). Перенесення сум у вигляді окремих операцій на акаунтах відправника і одержувача слугує для кількох важливих цілей:

1) Послідовність вхідних переказів, які по суті є асинхронними.
2) Зберегти транзакцію невеликою щоб поміститися в udp-пакети.
3) Полегшувати ledger обрізку шляхом мінімізації виведення даних.
4) Ізоляція прийнятих транзакцій від ще не прийнятих.

Більше одного аккаунта, що відправляє на один і той же є асинхронною операцією, затримка в мережі і відправляють акаунти не обов'язково повинні знаходитися в зв'язку один з одним, це означає, що немає універсального прийнятного способу дізнатися, яка транзакція сталася в першу чергу. Оскільки додавання асоціативне, порядок послідовних даних, що вводяться не має значення, і тому нам просто потрібно глобальна угода. Це ключовий компонент дизайну, який перетворює угоду про час виконання в угоду про час розробки. Приймаючий аккаунт має контроль над тим, щоб вирішити, який переказ прийшов першим і тим самим встановити свій порядок вхідних блоків.

Якщо аккаунт хоче зробити великий переказ, який був отриманий як набір невеликих переказів, ми хочемо представити це таким чином, який вписується в пакет UDP. Коли отримує аккаунт встановлює послідовність вхідних переказів, він зберігає загальну суму свого балансу рахунку, так що в будь-який час він мав можливість перевести будь-яку суму з фіксованим розміром транзакції. Це відрізняється від моделі транзакції введення / виведення використовуваної Bitcoin і іншими криптовалютами.

Деякі Ноди не зацікавлені в використанні ресурсів на зберігання повної історії транзакцій аккаунта; вони зацікавлені тільки в поточному балансі кожного аккаунта. Коли аккаунт здійснює транзакцію, він кодує накопичений баланс, і ці Ноди повинні відслідковувати тільки останній блок, який дозволяє їм відкидати історичні дані, зберігаючи при цьому вірність.

Навіть з акцентом на угоди про час розробки, є вікно затримки при перевірці транзакцій через виявлення і обробки поганих учасників в мережі. Так як угоди в RaiBlocks досягаються швидко, від мілісекунд до секунд, ми можемо представити користувача двом знайомим категоріям вхідних транзакцій: прийняті і не прийняті. Прийняті транзакції - це транзакції, в яких аккаунт створив блок отримання. Чи не прийняті транзакції ще не були включені до сукупного балансу одержувача. Це заміна більш складною і незвичною метрики підтвердження в інших криптовалютах.

Б. Створення Аккаунта

Щоб створити аккаунт, Вам необхідно створити Open транзакцію (Рис. 4). Open транзакція завжди є першою транзакцією кожного ланцюжка акаунтів і може бути створена при першому надходженні коштів. Поле account зберігає відкритий ключ (адреса), похідний від закритого ключа, який використовується для підпису. Поле source містить хеш транзакції, яка направила кошти. При створенні аккаунту, представник повинен бути обраний для голосування від вашого імені, він може бути змінений пізніше (розділ IV-E). Аккаунт може оголосити себе своїм представником.

```
open {
    account: DC04354B1...AE8FA2661B2,
    source: DC1E2B3F7C...182A0E26B4A,
    representative: xrb_1anr...posrs,
    work: 0000000000000000,
    type: open,
    signature: 83B0...006433265C7B204
}
```

Рис. 4. Приклад open транзакції

В. Баланс Аккаунта

Залишок на рахунку записується в ledger. Замість того щоб записувати суму транзакції, підтвердження (розділ IV-K) вимагає перевірки різниці між балансом в блоці відправки і балансом попереднього блоку. Одержувач аккаунт може потім збільшити попередній залишок, який вимірюється в кінцевому залишку, зазначеному в новому блоці прийому. Це робиться для підвищення швидкості обробки при завантаженні великих обсягів блоків. При запиті історії рахунку, суми вже дано.

Г. Відправлення з Аккаунта

Щоб відправити з адреси на адресу повинен бути вже відкритий блок, а також і баланс (Рис. 5). У previous полі міститься хеш попереднього блоку в ланцюжку аккаунта. Поле destination містить адресу для якого відправляються кошти. Блок відправки є незмінним після підтвердження. Після передачі в мережу кошти негайно віднімаються з балансу рахунку відправника та знаходяться в статусі pending для одержувача, поки він не підпише блок щоб прийняти ці кошти. Відкладені (Pending) кошти не слід вважати очікують підтвердження, оскільки вони вже витрачені з аккаунта відправника і відправник не може анулювати цю транзакцію.

```
send {
    previous: 1967EA355...F2F3E5BF801,
    balance: 010a8044a0...1d49289d88c,
    destination: xrb_3w...m37goeuufdp,
    work: 0000000000000000,
    type: send,
    signature: 83B0...006433265C7B204
}
```

Рис. 5. Приклад send транзакції

Д. Отримання Транзакції

Для завершення транзакції одержувач відправлених коштів повинен створити блок прийому на власному аккаунт-ланцюжку (Рис. 6). Поле source посилається на хеш транзакції відправки. Як тільки блок створений і трансльований в мережу, баланс аккаунта оновлюється і кошти офіційно зараховуються на рахунок одержувача.

```
receive {
    previous: DC04354B1...AE8FA2661B2,
    source: DC1E2B3F7C6...182A0E26B4A,
    work: 0000000000000000,
    type: receive,
    signature: 83B0...006433265C7B204
}
```

Рис. 6. Приклад receive транзакції

Е. Призначення представника (Representative)

Власники акаунтів, мають можливість вибрати представника для голосування від свого імені. Це є потужним інструментом децентралізації, яка не має сильного аналога в протоколі Proof of Work або Proof of Stake. У звичайних системах PoS у власника аккаунта повинна бути запущена нода для участі в голосуванні. Постійний запуск Ноди недоцільний для багатьох користувачів; передача представнику права голосу від імені аккаунта, що послаблює цю вимогу. Власники акаунтів мають можливість перепризначити представника аккаунта в будь-який час. Транзакція change змінює представника аккаунта, віднімаючи вагу голосування від старого представника і додавши вага для нового представника (рис. 7). Грошові кошти в даній операції не переміщуються, і представник не має можливості витрачати кошти на рахунок.

```
change {
    previous: DC04354B1...AE8FA2661B2,
    representative: xrb_1anrz...posrs,
    work: 0000000000000000,
    type: change,
    signature: 83B0...006433265C7B204
}
```

Рис. 7. Приклад change транзакції при зміні представника

Ж. Форк і Голосування

Форк виникає, коли j підписав блоки $b_1, b_2, \ldots, b_j$ изатвердив ті ж блоки, що і їх попередник (рис. 8). Ці блоки викликають суперечливі уявлення про стан аккаунта і повинні бути усунені. Тільки власник аккаунта має можливість підписувати блоки в свій ланцюжок акаунтів, тому форк повинен бути результатом поганого програмування або зловмисного наміру (подвійного витрачання) власником облікового запису.

Після виявлення представник створить голосування, що посилається на блок $\hat{b}_i$ в його ledger і передасть його в мережу. Вага голосування Ноди, w_i, є сумою балансів всіх рахунків, які назвали його своїм представником.

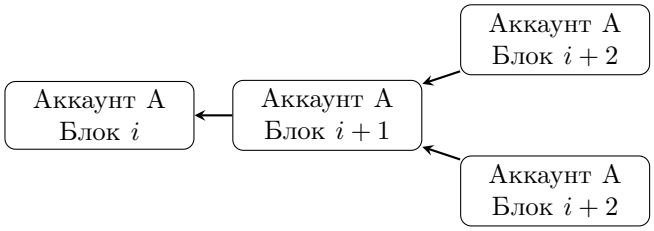


Рис. 8. Форк виникає, коли два (або більше) підписаних блоки посилаються на той же попередній блок. Старі блоки зліва, а нові блоки справа

Нода буде спостерігати чи входять голоси від інших M онлайн представників і зберігати накопичувальну відповіднсть на 4 періоди голосування, 1 хвилина на все, і підтвердить переможний блок (рівняння 1).

$$v(b_j) = \sum_{i=1}^{M} w_i \mathbb{1}_{\hat{b}_i = b_j} \tag{1}$$

$$b^* = \arg\max_{b_j} v(b_j) \tag{2}$$

Самий популярний блок b^* матиме більшість голосів і буде збережений в ledger Ноди (рівняння 2). Блок(и), що втратили голоси, видаляються. Якщо представник замінить блок у своїй ledger, він створить нове голосування з більш високим порядковим номером і передасть нове голосування в мережу. Це єдиний сценарій, коли голосують представники.

У деяких випадках короткі проблеми підключення до мережі можуть привести до того, що блок який передається не буде прийнятий усіма пірами (peers). Будь-який наступний блок на цьому акаунті буде проігнорований як неприпустимий пірами, які не бачили початкову передачу. Повторна трансляція цього блоку буде прийнята іншими пірами і наступні блоки будуть вилучені автоматично. Навіть при виникненні форку або відсутнього блоку зачіпаються тільки акаунти, зазначені в транзакції; інша частина мережі продовжує обробку транзакцій для всіх інших акаунтів.

И. Proof of Work

Всі чотири типи транзакцій мають робоче поле, яке повинно бути правильно заповнене. Поле work дозволяє творцю транзакції обчислити nonce таке, щоб хеш-код nonce об'єднувався з попереднім полем в receive/send/change транзакціях або в поле аккаунта у відкриій транзакції, нижче певного порогового значення. На відміну від Bitcoin, PoW в RaiBlocks використовується як анти-спам інструмент, аналогічна система Hashcash, і може бути обчислений за кілька секунд [9]. Після відправки транзакції PoW для подальшого блоку можливо попередньо обчислити, так як відомо попереднє поле блоку. Це робить транзакції миттєвими для кінцевого користувача до тих пір, поки час між транзакціями перевищує час, необхідний для обчислення PoW.

К. Перевірка Транзакцій

Щоб блок вважався припустимим, він повинен мати такі атрибути:

1) Блок вже не повинен бути в ledger (повторювані операції).
2) Повинен бути підписаний власником аккаунту.
3) Попередній блок є головним в ланцюжку аккаунта. Якщо він існує, але не головний, то це форк.
4) Аккаунт повинен мати блок відкриття.
5) Обчислений хеш відповідає пороговому значенню PoW.

Якщо це блок отримання, перевірте, чи очікує хеш вихідного блоку, тобто він ще не був погашений. Якщо це блок відправки, поточний баланс має бути менше попереднього значення балансу.

V. ВЕКТОРИ АТАК

RaiBlocks, як і всі децентралізовані криптовалюти, може піддатися нападу з боку зловмисників в спробі отримання фінансової вигоди або руйнування системи. У цьому розділі ми описуємо кілька можливих сценаріїв атаки, наслідки таких атак, і які превентивні заходи вживає протокол RaiBlock.

А. Синхронізація Блоків

В Розділі IV-Ж ми обговорили сценарій, коли блок може бути не переданий належним чином, в результаті чого мережа ігнорує наступні блоки. Якщо нода спостерігає за блоком, у якого немає зазначеного попереднього блоку, вона має два варіанти:

1) Ігнорувати блок, так як це може бути шкідливий сміттєвий блок.
2) Запит повторної синхронізації з іншої ноди.

У разі повторної синхронізації TCP-з'єднання повинно бути сформовано з завантажувальної ноди, щоб знизити збільшення обсягу трафіку, необхідного для повторної синхронізації. Однак, якщо блок насправді був поганим блоком, то повторна синхронізація є непотрібною, що веде до збільшення непотрібного трафіку в мережі. Ця атака на мережу і призводить до відмови в обслуговуванні.

Щоб уникнути непотрібної повторної синхронізації, Ноди чекатимуть, поки не буде досягнутий певний поріг голосів для потенційно шкідливого блоку, перш ніж ініціювати з'єднання з нодою для початку синхронізації. Якщо блок не отримує достатньої кількості голосів, його можна вважати небажаним.

Б. Флуд Транзакціями

Зловмисник може відправити багато непотрібних, але дійсних транзакцій між обліковим записами під його контролем, намагаючись наситити мережу. Без комісій за транзакції вони можуть продовжувати цю атаку дуже довго. Проте, PoW, необхідний для кожної транзакції, обмежує швидкість транзакції, яку може

створити зловмисна організація, без значного інвестування в обчислювальні ресурси. Навіть при такій атаці, намагаючись роздати ledger, Ноди, які НЕ вживають повну історію блоків, здатні обрізати старі транзакції зі свого ланцюжка, це убезпечить використання ledger від такого типу атаки майже для всіх користувачів.

B. Sybil Атака

Атакуючий може створити сотні нод RaiBlocks на одному комп'ютері, проте, оскільки система голосування заснована на балансі рахунків, додавання додаткових вузлів в мережу не дасть зловмиснику додаткових голосів. Тому немає ніяких переваг, який буде отримано через атаку Sybil.

Г. Атака пенні-витрати

Атака на пенні (копійками)- це те, де атакуючий витрачає нескінченно малі кошти на велику кількість акаунтів, щоб засмітити запам'ятовуючі пристрої нод. Публікації блоків обмежена по швидкості в PoW, тому це обмежує створення акаунтів і транзакцій в певній мірі. Ноди, які не є повними історичними вузлами, можуть обрізати акаунти нижче статистичної метрики, де аккаунт, швидше за все, не робітник. Нарешті, RaiBlocks налаштований на використання мінімального постійного простору для зберігання, тому простір, необхідне для зберігання одного додаткового аккаунта, пропорційний розміру відкритого блоку + індексування = 96B + 32B = 128B. Що дозволяє в 1 ГБ зберігати 8 мільйонів акаунтів за рахунок пенні. Якщо вузли захочуть обрізати більш агресивно, вони можуть розрахувати розподіл на основі частоти доступу і делегувати нечасто використовувані акаунти для більш повільного зберігання.

Д. Попередньо вирахувана атака PoW

Оскільки власник акаунта буде єдиною особою, що додає блоки в ланцюжок аккаунта, послідовні блоки можна обчислити разом з їх PoW, перш ніж передавати їх в мережу. Тут зловмисник генерує безліч послідовних блоків, кожен з яких має мінімальне значення, протягом тривалого періоду часу. У певний момент зловмисник виконає Denial of Service (DoS) шляхом флуда мережі з великою кількістю допустимих транзакцій, які інші Ноди будуть намагатися обробити якомога швидше. Це вдосконалена версія флуду в транзакціях, описаних в розділі V-Б. Така атака буде діяти недовго, але може використовуватися в поєднанні з іншими атаками такими як >50% Атаки (розділ V-E) для підвищення ефективності. В даний час розслідуються обмеження швидкості передачі і інші методи для полегшення атак.

E. >50% Атака

Показником консенсусу для RaiBlocks є виважена система голосування. Якщо зловмисник може набрати більше50% голосів, це може привести до того, що мережа буде коливатися в результаті збою системи. Зловмисник може знизити суму балансу, яку вони повинні втратити, не допускаючи, щоб хороші Ноди голосували використовуючи реалізацію мережевий DoS. RaiBlocks вживаються наступні заходи для запобігання такій атаки:

1) Первинний захист від такого типу атаки - це вага голосування, прив'язана до інвестицій в систему. Власник акаунта за своєю суттю стимулюється підтримувати чесність системи для захисту своїх інвестицій. Спроба змінити ledger буде руйнівною для системи в цілому, яка зруйнує і їх інвестиції.
2) Вартість такої атаки пропорційна ринкової капіталізації RaiBlocks. У системах PoW можна винайти технологію, яка дає непропорційний контроль в порівнянні з грошовими вкладеннями, і якщо атака буде успішною, цю технологію можливо направити на інші цілі. У RaiBlocks вартість атаки на систему залежить від самої ж системи і, якщо атака повинна бути успішною, інвестиції в атаку не можуть бути відновлені.
3) Щоб зберегти максимальний кворум голосуючих, наступна лінія захисту є репрезентативне голосування. Власники акаунтів, які не можуть брати участь в голосуванні з причин підключення до мережі, можуть призначити представника, який буде голосувати з вагою їх балансу. Максимізація числа представників підвищує стійкість мережі.
4) Форки в RaiBlocks ніколи не бувають випадковими, тому Ноди можуть приймати правила про те, як взаємодіяти з роздвоєними блоками. Єдиний раз, коли неатакуючі акаунти уразливі для блокування форк - якщо вони отримують баланс від атакуючого аккаунта. Акаунти, які хочуть бути захищеними від блокуючих форків, можуть почекати деякий час, перш ніж отримати від аккаунта, який згенерував форк, або ніколи не отримувати взагалі цей блок. Одержувачі також можуть створити окремі акаунти, які будуть використовуватися при отриманні коштів з сумнівних акаунтів, щоб ізолювати інші свої акаунти.
5) Останній захід захисту, який ще не реалізовано - block cementing. RaiBlocks робить все можливе, щоб швидко усунути блок-форк шляхом голосування. Ноди можуть бути налаштовані для block cementing, щоб запобігти їх відкат після певного періоду часу. Мережа досить захищена шляхом фокусування на швидкий час запобігання неоднозначних форків.

Складніша версія атаки $> 50\%$ детально описана на малюнку 9. «Offline» - це відсоток представників, які знаходяться не онлайн в момент голосування. «Stake» - це сума інвестиції, з якої зловмисник голосує. «Active» - це представники, які знаходяться в режимі онлайн і голосують за протоколом. Зловмисник може компенсувати суму, яку вони повинні втратити, вибивши

інших голосуючих в автономному режимі через мережеву DoS- атаку. Якщо ця атака стане стійкою, яких атакували представники стануть несинхронізованими і це демонструє «Unsync». Нарешті, зловмисник може отримати короткий розрив у відносній сили голосування, переключивши атаку DoS для нового набору представників, в той час як старий набір повторно синхронізує свій ledger, це демонструє «Attack».

Offline	Unsync	Attack	Active	Stake

Рис. 9. Потенційний механізм голосування, який може знизити вимоги до атаки на 51%.

Якщо зловмисник може викликати Stake> Active шляхом об'єднання цих обставин, він зможе успішно перекинути голоси в ledger за рахунок своєї частки. Ми можемо оцінити, наскільки цей тип атаки цінний, досліджуючи ринкову капіталізацію інших систем. Якщо ми припустимо, що 33% представників знаходяться в offline режимі або атаковані через DoS, зловмисникові необхідно буде купити 33% від ринкової капіталізації, щоб атакувати систему шляхом голосування.

Ж. Bootstrap Poisoning

Чим довше зловмисник буде зберігати старий закритий ключ з балансом, тим вище ймовірність того, що баланси, що існували в той час, не матимуть представників, тому що їхні баланси або представники перейшли на більш нові акаунти. Це означає, що якщо нода завантажує старого представника мережі, де зловмисник має більшість для голосування в порівнянні з представниками в той момент часу, вони зможуть варіювати рішення про голосування на цій ноді. Якщо цей новий користувач хоче взаємодіяти з будьким, крім атакуючої Ноди, всі його транзакції будуть відхилятися, так як вони мають різні великі блоки. Кінцевим результатом є те, що Ноди можуть витрачати час на нові вузли в мережі, передаючи їм погану інформацію. Щоб запобігти цьому, Ноди можуть бути пов'язані з вихідною базою даних облікових записів і з добре відомими головними блоками, це заміна для завантаження бази даних аж до блоку генезису. Чим ближче завантаження буде до актуальної версії, тим вище ймовірність повного захисту від цієї атаки. Зрештою, ця атака, ймовірно, не гірше, ніж завантаження небажаних даних в Ноди при завантаженні блоків, оскільки вони не зможуть здійснювати операції з будьким, хто має актуальну базу даних.

VI. РЕАЛІЗАЦІЯ

В даний час еталонна реалізація реалізована в C ++ і випускає релізи з 2014 року на Github [10].

А. Особливості дизайну

Реалізація RaiBlocks відповідає стандарту архітектури, описаного в цій статті. Додаткові характеристики описані тут.

1) Алгоритм підпису: RaiBlocks використовує модифікований алгоритм еліптичної кривої ED25519 з хешируванням Blake2b для всіх цифрових підписів [11]. ED25519 був обраний для швидкого підпису, швидкої перевірки і високої безпеки.

2) Алгоритм Хешірування: Оскільки алгоритм хешірування використовується тільки для запобігання спаму мережі, вибір алгоритму менш важливий в порівнянні з криптовалютою заснованих на майнингу. Наша реалізація використовує Blake2b як цифровий алгоритм проти вмісту блоку [12].

3) Функція деривації ключів: У гаманці ключі шифруються паролем, а пароль передається через функцію деривації ключів для захисту від спроб злому ASIC. В даний час Argon2 [13] є переможцем єдиного публічного конкурсу, спрямованого на створення відмовостійкої функції деривації ключів.

4) Блоковий йіинтервал: Оскільки кожен аккаунт має свій власний ланцюжок, оновлення можуть виконуватися асинхронно зі станом мережі. Тому інтервали між блоками немає і транзакції можуть бути опубліковані миттєво.

5) Протокол повідомленньй UDP: Наша система розрахована на безстрокову роботу з мінімальним обсягом обчислювальних ресурсів. Всі повідомлення в системі були розроблені таким чином, щоб вони були без стану і містилися в одному пакеті UDP. Це також полегшує для полегшених пірів з переривчастим підключенням участі в мережі без повторного відновлення короткострокових TCP-з'єднань. TCP використовується тільки для нових вузлів, коли вони хочуть завантажити ланцюжки блоків масовим способом.

Ноди можуть бути впевнені, що їх транзакції надійшли в мережу при дотриманні транзакцій широкомовного трафіку іншими нодамі, а це повинні побачити кілька копій відлунням повернувся до себе.

Б. Протокол IPv6 та multicast

Створення поверх UDP без встановлення з'єднання дозволяє в майбутньому використовувати багатоадресне розсилання по IPv6 в якості заміни традиційного флуда транзакцій і передачі голосувань. Це зменшить споживання пропускної здатності мережі і дасть більше гнучкості правил для нод.

В. Продуктивність

На момент написання цієї статті мережа RaiBlocks опрацювала 4,2 мільйона транзакцій, отримавши розмір блокчейна розміром 1,7 ГБ. Час транзакції вимірюється в секундах. Поточна еталонна реалізація, заміряна на SSD, може обробляти більше 10 000 транзакцій в секунду, в першу чергу це пов'язано з обмеженням IO.

VII. ВИКОРИСТАННЯ РЕСУРСІВ

Це огляд ресурсів, використовуваних нодою RaiBlocks. Крім того, ми переходимо до ідей зменшення використання ресурсів для конкретних випадків

використання. Спрощені Ноди зазвичай називаються легкими, обрізаними або спрощеними для верифікації платежу.

A. Мережа

Обсяг мережевої активності залежить від того, наскільки мережу сприяє здоров'ю мережі.

1) Представницька: Представницька нода вимагає максимальних мережевих ресурсів, оскільки вона стежить за трафіком голосів від інших представників і публікує свої власні голоси.

2) Без довіри: Недовірена нода схожа на представницьку ноду, але є тільки спостерігачем і не містить представницького закритого ключа акаунтів, а також не публікує власні голоси.

3) З довірою: Довірена нода спостерігає за трафіком голосів від одного представника, якому він довіряє, щоб правильно виконати консенсус. Це скорочує кількість вхідного трафіку голосування від представників, які посилаються на цю ноду.

4) Легка: Легка нода також є довіреною вузлом, який відстежує трафік тільки для акаунтів, в яких він зацікавлений, що дозволяє мінімізувати використання мережі.

5) Завантажувальна: Завантажувальна нода обслуговує частину або весь ledger для нод, які підключені до мережі. Це робиться через TCP-з'єднання, а не UDP, оскільки це передбачає велику кількість даних, що вимагає додаткового контролю потоку.

Б. Займане місце на диску

Залежно від вимог користувача різні конфігурації нод вимагають різних умов до зберігання.

1) Історична: Нода, зацікавлена в збереженні повної історії записів всіх транзакцій, зажадає максимального місця зберігання.

2) Поточнийй: У зв'язку з дизайном збереження накопичених балансів з блоками, Ноди повинні тримати тільки останні або головні блоки для кожного акаунта для того, щоб брати участь в консенсусі. Якщо нода не зацікавлена в збереженні повної історії, вона може зберігати тільки головні блоки.

3) Легка: Легка нода не зберігає дані з локального ledger, але бере участь тільки в мережі, щоб спостерігати за діяльністю на акаунтах, в яких вона зацікавлена, або, можливо, створювати нові транзакції з закритими ключами.

B. Процесор (CPU)

1) Генерація Транзакцій: Нода зацікавлена в створенні нових транзакцій, які повинні зробити Proof of Work nonce, щоб пройти механізм регулювання RaiBlocks. Обчислення різних апаратних засобів наведено в додатку A.

2) Представницька: Представник повинен перевіряти підписи для блоків, голосів, а також створювати свої власні підписи для участі в консенсусі. Обсяг використовуваних ресурсів CPU для представницької Ноди значно менше, ніж при генерації транзакції і може працювати з будь-яким процесором на сучасному комп'ютері.

3) Спостерігаюча: Спостережна нода не генерує свої власні голоси. Оскільки використовувані ресурси на підпис мінімальні, вимоги до процесора майже ідентичні роботі з представницької нодою.

VIII. ВИСНОВОК

У цьому документі ми представили фреймворк для довіреної, безкоміссіонной і з низькою затримкою криптовалюту, яка використовує нову структуру блокрешітки і делегованого PoS голосування. Мережа вимагає мінімальних ресурсів, не вимагає потужного устаткування для інтелектуального аналізу даних і може обробляти високу пропускну здатність транзакцій. Все це досягається за рахунок наявності окремих блоків для кожного акаунта, усунення проблем доступу і неефективності глобальної структури даних. Ми ідентифікували можливі атаки в системі і представили аргументи на ставлення того, наскільки RaiBlocks стійкий до цих форм атак.

Додаток A
POW АПАРАТНІ ПОКАЗНИКИ

Як згадувалося раніше, PoW в RaiBlocks - це для скорочення мережевого спаму. Наша реалізація нод забезпечує прискорення, яке може використовувати переваги сумісних з OpenCL графічних процесорів. В таблиці I наведено порівняння продуктивності різних апаратних засобів в реальних умовах. В даний час поріг PoW фіксований, але адаптивний поріг може бути реалізований в міру досягнення середньої обчислювальної потужності.

Табл. I
ПРОДУКТИВНІСТЬ ОБЛАДНАННЯ ДЛЯ POW

Device	Transactions Per Second
Nvidia Tesla V100 (AWS)	6.4
Nvidia Tesla P100 (Google,Cloud)	4.9
Nvidia Tesla K80 (Google,Cloud)	1.64
AMD RX 470 OC	1.59
Nvidia GTX 1060 3GB	1.25
Intel Core i7 4790K AVX2	0.33
Intel Core i7 4790K,WebAssembly (Firefox)	0.14
Google Cloud 4 vCores	0.14-0.16
ARM64 server 4 cores (Scaleway)	0.05-0.07

ПОДЯКА

Ми хотіли б подякувати Брайан Пью за компіляцію та форматування цієї статті.

Література

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: http://bitcoin.org/bitcoin.pdf

[2] “Bitcoin median transaction fee historical chart.” [Online]. Available: https://bitinfocharts.com/comparison/bitcoin-median_transaction_fee.html

[3] “Bitcoin average confirmation time.” [Online]. Available: https://blockchain.info/charts/avg-confirmation-time

[4] “Bitcoin energy consumption index.” [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption

[5] S. King and S. Nadal, “Ppcoin: Peer-to-peer crypto-currency with proof-of-stake,” 2012. [Online]. Available: https://peercoin.net/assets/paper/peercoin-paper.pdf

[6] C. LeMahieu, “Raiblocks distributed ledger network,” 2014.

[7] Y. Ribero and D. Raissar, “Dagcoin whitepaper,” 2015.

[8] S. Popov, “The tangle,” 2016.

[9] A. Back, “Hashcash - a denial of service counter-measure,” 2002. [Online]. Available: http://www.hashcash.org/papers/hashcash.pdf

[10] C. LeMahieu, “Raiblocks,” 2014. [Online]. Available: https://github.com/clemahieu/raiblocks

[11] D. J. Bernstein, N. Duif, T. Lange, P. Shwabe, and B.-Y. Yang, “High-speed high-security signatures,” 2011. [Online]. Available: http://ed25519.cr.yp.to/ed25519-20110926.pdf

[12] J.-P. Aumasson, S. Neves, Z. Wilcox-O’Hearn, and C. Winnerlein, “Blake2: Simpler, smaller, fast as md5,” 2012. [Online]. Available: https://blake2.net/blake2.pdf

[13] A. Biryukov, D. Dinu, and D. Khovratovich, “Argon2: The memory-hard function for password hashing and other applications,” 2015. [Online]. Available: https://password-hashing.net/argon2-specs.pdf