Одеський національний університет імені І. І. Мечникова
Факультет математики, фізики та інформаційних технологій
Кафедра математичного аналізу

Дипломна робота

бакалавра

на тему: **«Алгоритми захисту розподілених систем в блокчейн технологіях»**

«Protection algorithms of distributed systems in blockchain technologies»

Виконав: студент денної форми навчання
спеціальності 111 Математика
Гудков Олексій Миколайович

Керівник: доктор фіз.-мат. наук, проф.
Кореновський А. О.
Рецензент: кандидат фіз.-мат. наук, доц.
Вартанян Г. М.

Рекомендовано до захисту:
Протокол засідання кафедри
№ ___ від «_____» _____ р.
Завідувач кафедри
_____ _____

Захищено на засіданні ЕК № _____
Протокол № ____ від «____» ____ р.
Оцінка _____ / _____ / _____
Голова ЕК
_____ _____

Одеса — 2022 р.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі ми бачимо, що сфера блокчейну розвивається з величезною швидкістю. Ми розглянемо консенсуси Proof-of-Work та Proof-of-Stake на основі двох робіт: Bitcoin: A Peer-to-Peer Electronic Cash System, Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. Незважаючи на те, що системи, які працюють на технології Proof-of-Work застосовувалися вперше для розподілених систем та мають певні недоліки у своїй життедіяльності, протокол Proof-of-Stake прийшов щоб вирішити одну с його головних проблем, але Proof-of-Stake також має певні недоліки.

Мета та завдання роботи.

Метою роботи є вивчення двох консесусів.

1) Розгляд Proof-of-Work як перший консенсус, його проблеми, захист та перевірка платежу.
2) Розгляд Proof-of-Stake як інший погляд на рішення проблем Proof-of-Work блокчейну, огляд протоколу, розділення долі, розгалуження.

Мотивація. Кожна людина зараз має банківський рахунок. Коли вона робить транзакцію, ці гроші спочатку обробляються довіреним лицем у вігляді банку. У цей же час блокчейн пропонує рішення при якому люди зможуть обробляти транзакції без участі третіх осіб(банків). Також це рішення будет економити ресурси і мати менші комісії.

Почнемо з огляду консенсусу Proof-of-Work, а саме з його побудови. У роботі розглянуто кроки побудови протоколу, рішення проблем щодо захисту, запроваджено схеми забезпечення дискового простору, сценарії дій зловмисника та його варіанти взлому.

Далі ми розглянемо консенсус Proof-of-Stake та його особливості. Він віздрізняється від Proof-of-Work багатою кількістю речей, зокрема дизайном, виконанням та іншим. Proof-of-Stake відрізняється від тих, які приймають статичні повноваження, тим, що ставка змінюється з часом. Ця робота має реферативний характер і ми робимо огляд протоколів.

РОЗДІЛ 1

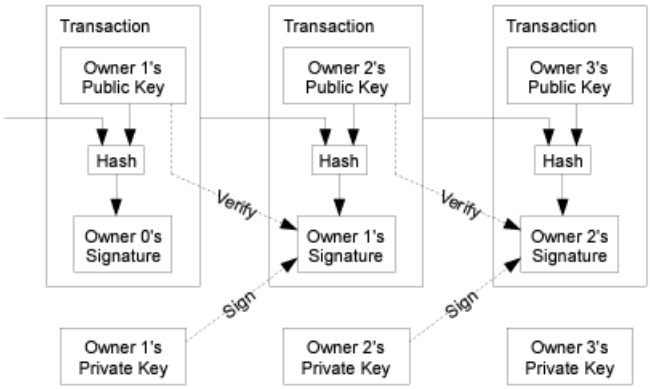
БІТКОІН ТА ЙОГО РЕЛІЗАЦІЯ

Вступ. Комерція в Інтернеті стала покладатися майже виключно на фінансові установи, які служать довіреними третіми сторонами для обробки електронних платежів. Сучасна система працює стабільно для більшості але, вона все ще страждає від внутрішніх недоліків моделі, заснованої на довірі. Незворотні операції насправді неможливі, оскільки фінансові установи не можуть уникнути посередницьких спорів. Вартість посередництва збільшує трансакційні витрати, обмежуючи мінімальний практичний розмір транзакції та відсікаючи можливість для невеликих випадкових транзакцій, а також є більш широкі витрати у втраті можливості здійснювати незворотні платежі. Зворотна транзакція потребує рівня довіри. Продавці повинні обережно ставитися до своїх клієнтів, домагаючись від них більше інформації, ніж їм було б потрібно. Шахрайство буде неминучим у деякому відсотку. Витрат і невизначеності платежів можна уникнути за допомогою використання фізичної валюти, але не існує механізму здійснення платежів через канал зв'язку без довіреної сторони. Потрібна електронна платіжна система, заснована на криптографічному доказі, а не на довірі, що дозволяє будь-яким двом бажаючим сторонам здійснювати транзакції безпосередньо один з одним без потреби довіреної третьої сторони. Транзакції, які неможливо відмінити в обчислювальному відношенні, захистили б продавців від шахрайства, а для захисту покупців можна було б легко запровадити рутинні механізми депонування. Ми розглядаємо рішення проблеми подвійних витрат за допомогою однорангового розподіленого сервера міток часу для створення обчислювального підтвердження хронологічного порядку транзакцій. Система безпечна до тих пір, поки чесні вузли спільно контролюють більше потужності ЦП, ніж будь-яка група співпрацюючих вузлів-зловмисників.

1.1. Транзакції та сервер міток часу

Транзакції

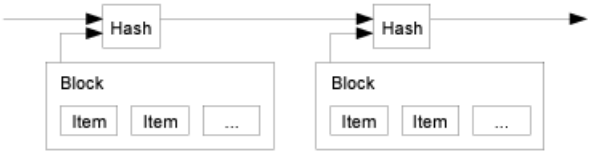
Почнемо з опису транзакцій та пояснення як все це повинно працювати. Ми визначаємо **електронну монету** як ланцюжок цифрових підписів. Кожен власник передає монету наступному шляхом цифрового підпису хеша попередньої транзакції та відкритого ключа наступного власника та додавання їх у кінець монети. Одержувач може перевірити підписи, щоб перевірити ланцюжок власності.



Проблема полягає в тому, що одержувач не може перевірити подвійну витрату монети. Поширеним рішенням є створення надійного центрального органу, або монетного двору, який перевіряє кожну транзакцію на подвійні витрати. Після кожної транзакції монета повинна повернутися на монетний двір і повинна буде випущена нова, тоді монети будуть витрачатися двічі. Проблема з цим рішенням полягає в тому, що доля всієї грошової системи залежить від компанії, яка керує монетним двором, і кожна транзакція повинна проходити через них, як і банк. Нам потрібен спосіб, щоб одержувач знав, що попередні власники не підписували жодних попередніх угод. Для наших цілей важлива сама рання транзакція, тому ми не піклуємося про подальші спроби подвійного витрачання. Знати про відсутність транзакції ми зможемо лише при умові знання про всі транзакції. У моделі на основі монетного двору, монетний двір був в курсі всіх транзакцій і вирішував, які з них прибули першими. Щоб виконати це без довіреної сторони, транзакції мають бути публічно оголошені, і нам потрібна система, щоб учасники погоджували єдину історію порядку їх отримання. При виповненні транзакції одержувачу потрібен доказ того, що це була перша транзакція, де більшість вузлів погодились.

Сервер міток часу

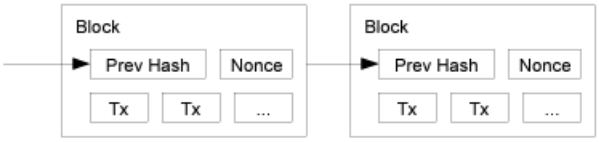
Рішення, яке пропонується, починається з сервера часових позначок. Сервер міток часу працює так: бере хеш блоку елементів, які мають бути позначені часовою міткою, і широко публікує хеш, наприклад, у газеті чи дописі Usenet. Позначка часу доводить, що дані повинні існувати на той момент, щоб потрапити в хеш. Кожна позначка часу включає попередню позначку часу в свій хеш, утворюючи ланцюжок, причому кожна додаткова мітка часу посилює попередні.



1.2. Доказ роботи, запуск мережи, мотивація

Доказ роботи(Proof-of-Work)

Щоб реалізувати розподілений сервер міток часу на основі одноранго-вого зв'язку, потрібно буде використовувати систему Proof-of-Work, подібну до Hashcash Адама Бека. Підтвердження роботи включає в себе скануван-ня значення, яке при хешуванні, як от SHA-256, починається з кількості нульових бітів. Середня необхідна робота є експоненційною щодо кількості необхідних нульових бітів і може бути підтверджена виконанням одного хе-шування. Для цієї мережі з мітками часу реалізуємо підтвердження роботи, збільшуючи одноразовий номер у блоці, доки не буде знайдено значення, яке надає хешу блоку необхідні нульові біти. Після того, як зусилля ЦП були витрачені, щоб він задовольнив підтвердження роботи, блок не мо-жна змінити без повторного виконання роботи. Оскільки наступні блоки об'єднуються в ланцюжок після нього, робота зі зміни блоку включатиме повторне виконання всіх блоків після нього.



Доказ роботи також вирішує проблему визначення представництва при прийнятті рішень більшості. Якби більшість базувалося на одній IP-адресі одного голосу, її міг би зруйнувати будь-хто, хто може видалити багато IP-адрес. По суті, підтвердження роботи — це один голос. Рішення більшості представлено найдовшим ланцюгом, у який вкладено найбільше зусиль для підтвердження роботи. Якщо більша частина потужності ЦП контролюється чесними вузлами, чесний ланцюжок буде рости найшвидше і випереджати будь-які конкуруючі ланцюги. Щоб змінити попередній блок, зловмиснику доведеться повторити підтвердження роботи блоку та всіх блоків після нього, а потім наздогнати та перевершити роботу чесних вузлів. Щоб компенсувати збільшення швидкості обладнання та зміну інтересу до запущених вузлів з часом, складність підтвердження роботи визначається ковзним середнім, орієнтованим на середню кількість блоків на годину. Якщо вони генеруються занадто швидко, складність збільшується.

Запуск мережи

Нижче наведено кроки для запуску мережі:

- Нові транзакції транслюються на всі вузли.
- Кожен вузол збирає нові транзакції в блок.
- Кожен вузол працює над пошуком складного підтвердження роботи для свого блоку.
- Коли вузол знаходить підтвердження роботи, він передає блок усім вузлам.
- Вузли приймають блок, тільки якщо всі транзакції в ньому дійсні і ще не витрачені.
- Вузли виражають своє прийняття блоку, працюючи над створенням наступного блоку в ланцюжку, використовуючи хеш прийнятого блоку як попередній хеш.

Вузли завжди вважають найдовший ланцюжок правильним і будуть

продовжувати працювати над його розширенням. Якщо два вузли одночасно транслюють різні версії наступного блоку, деякі вузли можуть отримати ту чи іншу першими. У цьому випадку вони працюють над першою, яку отримали, але зберігають іншу гілку, якщо вона стане довшою. Зв'язок буде розірваний, коли буде знайдена наступний вузел Proof-of-Work, і одна гілка стане довшою; вузли, які працювали на іншій гілці, потім перемикаються на довшу. Нові транзакції не обов'язково повинні охоплювати всі вузли. Поки вони досягають багатьох вузлів, вони незабаром потраплять у блок. Блокові трансляції також толерантні до пропущених повідомлень. Якщо вузол не отримує блок, він запитує його, коли отримає наступний блок і розуміє, що пропустив його.

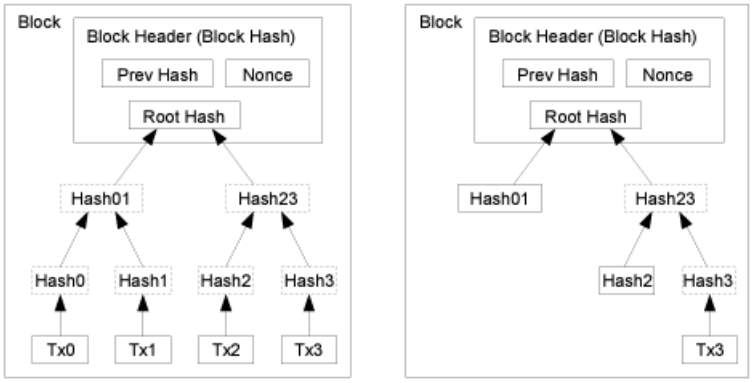
Мотивація підтримки мережи

За умовою, перша транзакція в блоці є спеціальною. Вона запускає нову монету, що належить творцю блоку. Це додає стимул для вузлів підтримувати мережу та надає спосіб початкового розповсюдження монет в обіг, оскільки немає центрального органу, який би їх випускав. Постійне додавання деякої кількості нових монет аналогічно тому, як золотошукачі витрачають ресурси, щоб додати золото в обіг. У цьому випадку витрачається процесорний час і електроенергія. Мотивація також може фінансуватися за рахунок комісій за транзакції. Якщо вихідне значення транзакції менше її вхідного значення, різниця є платою за транзакцію, яка додається до стимулюючого значення блоку, що містить транзакцію. Після того, як заздалегідь визначена кількість монет надійшла в обіг, стимул може повністю перейти на комісію за транзакції та повністю вільний від інфляції. Стимулювання може допомогти залишати вузли чесними. Якщо жадібний зловмисник може зібрати більше потужності процесора, ніж усі чесні вузли, йому доведеться вибирати між використанням цього для обману людей, викравши його платежі, або використанням для створення нових монет. Йому повинно бути вигідніше грати за правилами, які дають йому більше нових монет, ніж підривати систему та дійсність його власного багатства.

1.3. Дисковий простір, спрощена перевірка платежу

Відновлення дискового простору

Після того, як остання транзакція в монеті похована під достатньою кількістю блоків, витрачені транзакції перед нею можна буде відкинути, щоб заощадити місце на диску. Для полегшення, транзакції хешуються в дереві Меркла, не порушуючи хеш блоку, при цьому в хеш блоку входить лише корінь. Після цього ми можемо відколити гілки дерева для ущільнення блоків. Внутрішні хеші не потрібно зберігати.

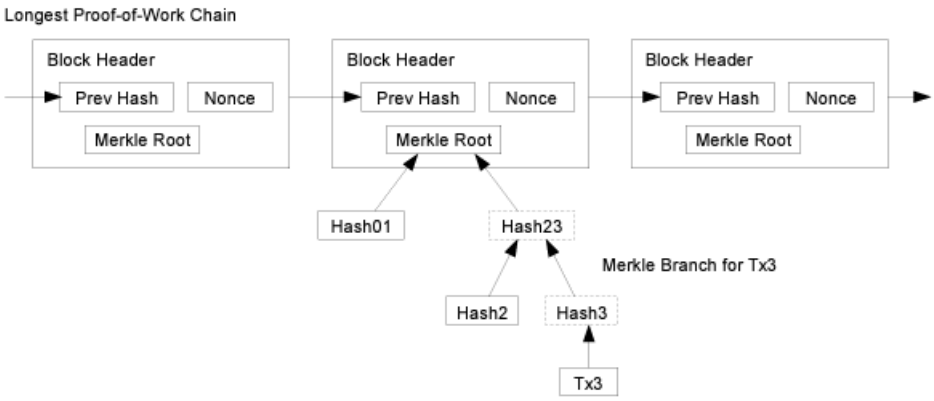


Transactions Hashed in a Merkle Tree — After Pruning Tx0-2 from the Block

Заголовок блоку без транзакцій буде приблизно 80 байт. Якщо припустити, що блоки генеруються кожні 10 хвилин, 80 байт * 6 * 24 * 365 = 4,2 МБ на рік. Оскільки комп'ютерні системи зазвичай продаються з 2 ГБ оперативної пам'яті станом на 2008 рік, а закон Мура передбачає поточне зростання на 1,2 ГБ на рік, сховище не повинно бути проблемою, навіть якщо заголовки блоків потрібно зберігати в пам'яті.

Спрощена перевірка платежу

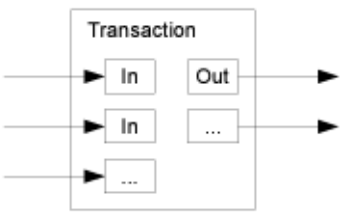
Можна перевірити платежі без запуску повного мережевого вузла. Користувачу потрібно лише зберегти копію заголовків блоків найдовшого ланцюга підтвердження роботи, яку він може отримати, запитуючи вузли мережі, поки не переконається, що він має найдовший Proof-of-Work ланцюжок, і отримати гілку Меркла, яка пов'язує транзакцію з блоком. Він не може перевірити транзакцію самостійно, але зв'язавши її з місцем у ланцюжку, він може побачити, що мережевий вузол прийняв її, і блоки, додані після того, як він додатково підтвердить, що мережа прийняла її.



Таким чином, перевірка є надійною, якщо мережу контролюють чесні вузли, але є більш вразливою якщо мережу подолає зловмисник. Хоча вузли мережі можуть самостійно перевіряти транзакції, спрощений метод може бути обдурений сфабрикованими транзакціями зловмисника до тих пір, поки зловмисник може продовжувати долати мережу. Однією зі стратегій захисту від цього було б приймати сповіщення від мережевих вузлів, коли вони виявляють недійсний блок, що спонукає програмне забезпечення користувача завантажити повний блок і сповіщати про транзакції для підтвердження невідповідності. Компанії, які отримують постійні платежі, ймовірно, захочуть запустити власні вузли для більш незалежної безпеки та швидшої перевірки.

Об'єднання та розщеплення значення

Звичайно, можна було б обробляти монети окремо, але ця операція буде надто витратна для обробки кожного цента при переказі. Щоб значення можна було розділити та об'єднати, транзакції містять кілька входів і виходів. Зазвичай буде або один вхід від більшої попередньої транзакції, або кілька введених даних, що об'єднують менші суми, і щонайбільше два вихідні дані: один для платежу, а другий повертає зміну, якщо така є назад відправнику.

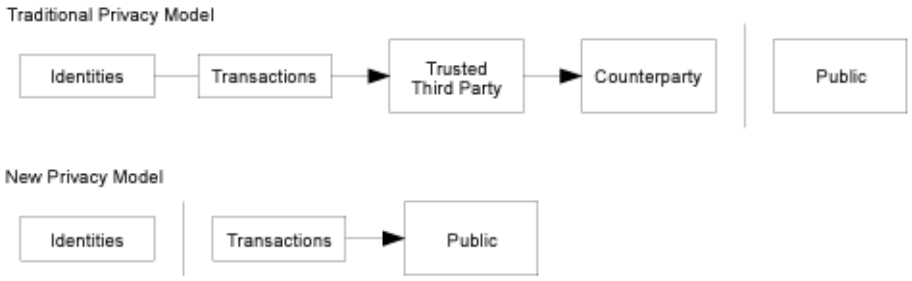


Слід зазначити, що коли транзакція залежить від кількох транзакцій, а ці транзакції залежать від багатьох інших, не є проблемою. Ніколи не потрібно витягувати повну окрему копію історії транзакцій.

1.4. Конфіденційність та розрахунки

Конфіденційність

Традиційна банківська модель забезпечує певний рівень конфіденційності, обмежуючи доступ до інформації залученими сторонами та довіреною третьою стороною. Необхідність публічно оголошувати всі транзакції виключає цей метод, але конфіденційність все ще можна підтримувати, перериваючи потік інформації в іншому місці: методом зберыгання анонімних відкритих ключів. Громадськість може бачити, що хтось надсилає суму комусь іншому, але без інформації, яка пов'язує транзакцію з кимось. Це схоже на інформацію на фондових біржах, яка оприлюднюється без вказування сторін угоди.



Як додатковий брандмауер для кожної транзакції слід використовувати нову пару ключів, щоб вони не були пов'язані зі загальним власником. Деякі зв'язки все ще неминучі з транзакціями з кількома вводами, які обов'язково показують, що їхні вхідні дані належали одному власнику. Основний ризик являється у виявленні власників ключів, який може провести до інших транзакцій власника.

Розрахунки

Розглянемо сценарій, коли зловмисник намагається створити альтернативний ланцюжок швидше, ніж чесний ланцюг. Навіть якщо це досягнуто,

це не відкриє систему для довільних змін, таких як створення цінності з повітря або вилучення грошей, які ніколи не належали зловмиснику. Вузли не приймуть недійсну транзакцію як платіж, а чесні вузли ніколи не приймуть блок, що містить їх. Зловмисник може намагтися повернути свої гроші, які він використав. Гонку між чесним ланцюгом і ланцюгом зловмисників можна охарактеризувати як біноміальну випадкову прогулянку. Подія успіху — це подовження чесного ланцюга на один блок, що збільшує його перевагу на $+1$, а подія невдачі — це подовження ланцюга зловмисника на один блок, що зменшує розрив на -1. Припускається, що гравець з необмеженим кредитом починає з дефіцитом і потенційно грає нескінченну кількість спроб, щоб спробувати вийти на беззбитковість. Можна розрахувати ймовірність того, що він коли-небудь досягне беззбитковості або що зловмисник коли-небудь наздожене чесний ланцюг, так:

p = ймовірність, що чесний вузол знайде наступний блок
q = ймовірність того, що зловмисник знайде наступний блок
q_z = ймовірність, що зловмисник коли-небудь наздожене від z блоків позаду

$$q_z = \begin{cases} 1 & \text{якщо} \quad p \leq q \\ (q/p)^z & \text{якщо} \quad p > q \end{cases}$$

Враховуючи припущення, що $p > q$, ймовірність падає експоненціально, оскільки зростає кількість блоків, які зловмисник повинен наздогнати. Враховуючи шанси проти нього, якщо він не зробить щасливий випад на початку, його шанси стають зникаючими, оскільки він відстає далі. Тепер розглянемо, скільки часу потрібно чекати одержувачу нової транзакції, перш ніж бути достатньо впевненим, що відправник не зможе змінити транзакцію. Припускаємо, що відправник є зловмисником, який хоче змусити одержувача повірити, що він заплатив йому, а потім переключити його на повернення собі через деякий час. Коли це станеться, одержувач отримає сповіщення, але відправник сподівається, що буде надто пізно. Одержувач створює нову пару ключів і надає відкритий ключ відправнику незадовго до підписання. Це не дозволяє відправнику заздалегідь підготувати ланцюжок блоків, працюючи над ним безперервно, поки йому не пощастить

зайти достатньо далеко вперед, а потім виконати транзакцію в цей момент. Після відправлення транзакції нечесний відправник починає таємно працювати над паралельним ланцюжком, що містить альтернативну версію його транзакції. Одержувач чекає, поки транзакція не буде додана до блоку, а після неї будуть пов'язані z-блоки. Він не знає точної кількості прогресу, досягнутого зловмисником, але якщо припустити, що чесні блоки зайняли середній очікуваний час на блок, потенційний прогрес зловмисника буде розподілом Пуассона з очікуваним значенням:

$$\lambda = z\frac{q}{p}$$

Щоб отримати ймовірність того, що зловмисник все ще зможе наздогнати, помножимо щільність Пуассона для кожної кількості прогресу, яку він міг би досягти, на ймовірність, яку він зможе наздогнати з цього моменту:

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \begin{cases} (q/p)^{z-k} & \text{якщо} \quad k \leq z \\ 1 & \text{якщо} \quad k > z \end{cases}$$

Переупорядкування, щоб уникнути підсумовування нескінченного хвоста розподілу

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!}(1 - (q/p)^{z-k})$$

У описі біткоіна запропоновано систему для електронних транзакцій без довіри. Почали зі звичайного каркаса монет, зробленого з цифрових підписів, який забезпечує жорсткий контроль власності, але є неповним без способу запобігання подвійних витрат. Щоб вирішити цю проблему, запропоновано однорангову мережу, яка використовує Proof-of-Work для запису загальнодоступної історії транзакцій, яка швидко стає обчислювально непрактичною для зловмисника, щоб змінити її, якщо чесні вузли контролюють більшу частину потужності ЦП. Мережа надійна у своїй неструктурованій простоті. Вузли працюють одночасно з невеликою координацією. Їх не

потрібно ідентифікувати, оскільки повідомлення не направляються в будь-яке конкретне місце, а доставляються лише з максимальними зусиллями. Вузли можуть залишати та знову приєднуватися до мережі за бажанням, приймаючи ланцюжок підтвердження роботи як доказ того, що сталося, коли їх не було. Вони голосують потужністю ЦП, виражаючи своє прийняття дійсних блоків, працюючи над їх розширенням, і відхиляючи недійсні блоки, відмовляючись працювати з ними. За допомогою цього механізму консенсусу можна застосувати будь-які необхідні правила та стимули.

РОЗДІЛ 2

PROOF-OF-STAKE ТА ЙОГО РЕАЛІЗАЦІЯ

Вступ. Основним ресурсом протоколів блокчейну Proof-of-Work є енергія необхідна для виконання. Для створення одного блоку в біткойн-блокчейні використовується кількість операцій хешування, що перевищує 2^{60}, що призводить до вражаючого споживання енергії. Справді, ранні розрахунки показали, що енергетичні вимоги протоколу були порівнянними з потребами невеликої країни. Такий стан справ спонукав до дослідження альтернативних протоколів блокчейну, які позбавили б необхідності підтвердження роботи, замінивши його іншим, більш енергоефективним механізмом, який може надати подібні гарантії.

Механізм підтвердження роботи біткойна сприяє типу рандомізованого процесу «виборів лідера», який обирає одного з майнерів для випуску наступного блоку. За умови, що всі майнери дотримуються протоколу, цей вибір виконується рандомізованим способом пропорційно обчислювальної потужності кожного майнера. (Відхилення від протоколу можуть спотворити цю пропорційність, як приклад стратегії «егоїстичного майнінгу».)

Природний альтернативний механізм спирається на поняття «доказ долі» (Proof-of-Stake). Майнери у біткоін-блокчейні інвестують свої ресурси для участі у процесі вибора лідера, коли у Proof-of-Stake процес випадковим чином вибирає одного пропорційно долі, якою він володіє. По суті, це створює дисципліну блокчейну з самореферентом: підтримка блокчейну залежить від самих зацікавлених сторін і призначає їм роботу (а також винагороду) на основі суми частки, якою кожен володіє, як зазначено в книзі. Крім цього, дисципліна не повинна пред'являти додаткових «штучних» обчислювальних вимог до зацікавлених сторін. Реалізація такого протоколу пов'язана з рядом дефініційних, технічних та аналітичних проблем.

Дизайн блокчейну на основі доказу участі більш формально вивчався Бентовим та інші, як у поєднанні з Proof-of-Work, а також єдиним механізмом для блокчейн протоколу. Хоча Бентов та інші показали, що їхні протоколи захищені від деяких класів атак, вони не надають формальної

моделі для аналізу протоколів на основі Proof-of-Stake або доказів безпеки, що спираються на точні визначення. Протоколи блокчейну на основі евристичного підтвердження ставок були запропоновані і реалізовані для низки криптовалют. Цікаво порівняти протокол блокчейну на основі Proof-of-Stake із класичним консенсусним блокчейном, який спирається на фіксований набір повноважень. Блокчейн Proof-of-Stake має ставку, що змінюється з часом та припущення довіри розвивається разом із системою у той час, коли інші приймають статичні повноваження.

Дизайнерський виклик Proof-of-Stake. Фундаментальною проблемою для протоколів блокчейн на основі Proof-of-Stake є моделювання процесу виборів лідера. Для досягнення чесних рандомізованих виборів серед зацікавлених сторін необхідно ввести ентропію в систему, а механізми введення ентропії можуть бути схильні до маніпуляцій з боку супротивника. Наприклад, супротивник, який контролює набір зацікавлених сторін, може спробувати змоделювати виконання протоколу, пробуючи різні послідовності учасників зацікавлених сторін, щоб знайти продовження протоколу, яке сприяє зацікавленим сторонам. Це призводить до так званої «шліфованої» вразливості, коли ворогуючі сторони можуть використовувати обчислювальні ресурси для упередження виборів лідера.

«Уроборос» — доказово безпечна система доведення ставок. Відомо, що це перший блокчейн-протокол такого роду з ретельним аналізом безпеки. Маємо:

По-перше, розглянемо модель, яка формалізує проблему реалізації протоколу блокчейн на основі Proof-of-Stake. Модель, яка вводиться, зосереджена на стійкості та живучості двох формальних властивостях надійної книги транзакцій. Постійність стверджує, що як тільки вузол системи оголошує певну транзакцію «стабільною», інші вузли, якщо запитували та відповідали чесно, також повідомлять про неї як про стабільну. Тут стабільність слід розуміти як предикат, який буде параметризовано деяким параметром безпеки k, що вплине на впевненість, з якою властивість зберігається. (Наприклад, «понад k блоків у глибину».) Живість гарантує, що після того, як чесно згенерована транзакція стане доступною для вузлів мережі протягом достатньої кількості часу, скажімо, u кроків часу, вона

стане стабільною. Поєднання живості та стійкості забезпечує надійну книгу транзакцій у тому сенсі, що чесно згенеровані транзакції приймаються і стають незмінними. Модель має відповідні зміни, щоб полегшити динаміку на основі Proof-of-Stake.

По-друге, описується новий протокол блокчейн на основі консенсусу Proof-of-Stake. Протокол передбачає, що сторони можуть вільно створювати рахунки, отримувати й здійснювати платежі, а ставка змінюється з часом. Реалізується безпечна багатопартійна реалізація протоколу підкидання монет, щоб створити випадковість для процесу виборів лідера. Це відрізняє наш підхід і запобігає «шліфуючим атакам» від інших попередніх рішень, які або визначали такі значення детерміновано на основі поточного стану блокчейну, або використовували колективне підкидання монет як спосіб введення ентропії. Крім того, унікальним для нашого підходу є той факт, що система ігнорує покрокові зміни ставки. Натомість знімок поточного набору зацікавлених сторін робиться через регулярні проміжки часу, які називають епохами. У кожному такому інтервалі виконується безпечне багатостороннє обчислення з використанням самого блокчейна як каналу широкомовної передачі. В кожну епоху набір випадково відібраних зацікавлених сторін формує комітет, який потім відповідає за виконання протоколу підкидання монет. Результат протоколу визначає набір наступних зацікавлених сторін для виконання протоколу в наступну епоху, а також результати всіх виборів лідера для цієї епохи.

По-третє, надається набір формальних аргументів, які встановлюють, що жоден супротивник не може зламати наполегливість і живість. Цей протокол безпечний за низки правдоподібних припущень: мережа є синхронною в тому сенсі, що може бути визначена верхня межа, протягом якої будь-яка чесна зацікавлена сторона може спілкуватися з будь-якою іншою зацікавленою стороною, кількість зацікавлених сторін, зібраних із чесної більшості, доступна для участі в кожній епосі, зацікавлені сторони не залишаються поза мережею протягом тривалого періоду часу, адаптивність корупції має невелику затримку, яка вимірюється в раундах, лінійних за параметром безпеки (або, як альтернатива, гравці мають доступ до каналу анонімного мовлення відправника). В основі аргументів щодо безпеки лежить імовірнісний аргумент щодо комбінаторного поняття «розгалужених рядків», яке

формулюється, доводиться, а також перевіряється експериментально. У аналізі також виділяємо приховані атаки, особливий клас загальних атак розгалуження. «Прихованість» тут інтерпретується в дусі прихованих супротивників проти безпечних багатосторонніх обчислювальних протоколів, де супротивник бажає порушити протокол, але вважає за краще, щоб його не спіймали на цьому. Показуємо, що струни, що приховано розщеплюються, є підкласом рядків, які можна розщелкувати, з набагато меншою щільністю; це дозволяє надати два різних аргументи безпеки, які досягають різних компромісів з точки зору ефективності та гарантій безпеки. Аналіз рядків із можливістю розгалуження — це природний і досить загальний інструмент, який можна застосувати як частину аргументу безпеки — налаштування Proof-of-Stake.

По-четверте, звертається увага на мотиваційну структуру протоколу. Представляємо новий механізм винагороди для стимулювання учасників до системи, яка, як виявляється, є (приблизною) рівновагою Неша. Таким чином, дизайн пом'якшує такі атаки, як блокування та егоїстичний майнінг. Основна ідея механізму винагороди полягає в тому, щоб забезпечити позитивну винагороду за ті протокольні дії, які не можуть бути задушені коаліцією партій, що відхиляються від протоколу. Таким чином можна показати, що за правдоподібних припущень, а саме, що витрати на виконання певного протоколу малі, точне дотримання протоколу є рівновагою, коли всі гравці є раціональними.

По-п'яте, вводимо механізм делегування участі, який можна легко додати до цього протоколу блокчейн. Делегування особливо корисно в контексті, оскільки хочеться щоб дозволялося нашому протоколу масштабуватися навіть в умовах, де набір зацікавлених сторін дуже фрагментований. У таких випадках механізм делегування може дозволити зацікавленим сторонам делегувати свої «права голосу», тобто право брати участь у роботі комітетів, які керують протоколом відбору лідера в кожну епоху. Як і в ліквідній демократії (вона ж «делегативна демократія»), зацікавлені сторони мають можливість скасувати своє призначення уповноважених, якщо вони бажають незалежно один від одного.

Враховуючи таку модель та опис протоколу, також досліджуємо,

як різні атаки, які розглядаються на практиці, можна подолати в нашій структурі. Зокрема, обговорюємо атаки подвійних витрат, атаки відмови у транзакціях, атаки 51%, атаки «нічого на карті», атаки десинхронізації та інші. Представляються докази ефективності дизайну. Спочатку розглянемо атаки подвійних витрат. Для ілюстрації проводимо порівняння з аналізом Накамото для біткойна щодо часу підтвердження транзакції з впевненістю 99,9%. Проти прихованих супротивників час підтвердження транзакції від 10 до 16 разів швидше, ніж у біткойна, залежно від потужності хешування з боку суперника; для загальних супротивників час підтвердження в 5-10 разів швидше.

Конкретний аналіз атак з подвійними витратами ґрунтується на комбінаторному аналізі рядків, які можна розщелкувати та приховано роздвоювати, і застосовується до набагато ширшого класу змагальної поведінки, ніж більш спрощений аналіз Накамото. Потім розглядаємо реалізацію прототипу та звітуємо про проведені контрольні експерименти в хмарі Amazon, які демонструють силу протоколу доказу частки блокчейн з точки зору продуктивності. Через брак місця представляємо вищевказане в повній версії.

2.1. Модель та протокол

Модель

Час, слоти та синхронізація. Налаштування, у якому час ділиться на дискретні одиниці, називається слотами. Головна книга, більш детально описана нижче, поєднується з кожним часовим інтервалом (максимум) одним блоком книги. Гравці оснащені (приблизно синхронізованими) годинниками, які вказують поточний слот. Це дозволить їм виконувати розподілений протокол, який має намір спільно призначити блок цьому поточному слоту. Загалом, кожен слот sl_r індексується цілим числом $r \in \{1,2,...\}$, і припускається, що вікно реального часу, яке відповідає кожному слоту, має такі властивості:

– Поточний слот визначається загальновідомою і монотонно зростаючою функцією поточного часу.

– Кожен гравець має доступ до поточного часу. Будь-які розбіжності між місцевим часом сторін незначні в порівнянні з проміжком часу, представленим слотом.
– Довжина часового вікна, що відповідає слоту, є достатньою для гарантії того, що будь-яке повідомлення, передане чесною стороною на початку часового вікна, буде отримано будь-якою іншою чесною стороною до кінця цього періоду часу (навіть з урахуванням незначних невідповідності місцевих годинників партій). Зокрема, хоча затримки в мережі можуть виникати, вони ніколи не перевищують часове вікно слоту.

Властивості книги транзакцій. Протокол P реалізує надійну книгу транзакцій за умови, що книга, яку веде P, розділена на «блоки» (призначені часовим інтервалам), які визначають порядок, у якому транзакції включаються в книгу. Він також повинен задовольняти двом наступним властивостям:

– **Наполегливість.** Після того, як вузол системи оголошує певну транзакцію tx стабільною, інші вузли, якщо запитують, або повідомляють tx на тій самій позиції в книзі, або не повідомляють як стабільну, жодну транзакцію, яка конфліктує з tx. Тут поняття стабільності є предикатом, який параметризується параметром безпеки k. Зокрема, транзакція оголошується стабільною тоді і тільки тоді, коли вона знаходиться в блоці, що містить більше k блоків у реєстрі.
– **Живучість.** Якщо всі чесні вузли в системі намагаються включити певну транзакцію, то після закінчення часу, що відповідає слотам u (називається часом підтвердження транзакції), усі вузли, якщо запитували та відповідали чесно, скажуть про транзакцію як про стабільну.
– **Загальний префікс (CP);** з параметрами $k \in N$. Ланцюжки C_1, C_2, якими володіють дві чесні сторони на початку слотів $sl_1 < sl_2$, такі, що $C_1^k \leq C_2$ позначає ланцюг, де C_1^k отриманий шляхом видалення останніх k блоків із C_1, і $\leq$ позначає префіксне відношення.
– **Якість ланцюга (CQ);** з параметрами $\mu \in (0,1] \to (0,1]$ та $l \in N$. Розглянемо будь-яку частину довжини принаймні ланцюга, якою володіє чесна сторона на початку раунду; відношення блоків, що виходять

від супротивника, не більше $1-\mu$. Називається μ коефіцієнтом якості ланцюга.

– **Зростання ланцюга (CG);** з параметрами $\tau \in (0,1] \to (0,1]$ та $s \in N$. Розглянемо ланцюги C_1, C_2, якими володіють дві чесні сторони на початку двох слотів sl_1, sl_2 з sl_2 щонайменше s слотами попереду sl_1. Тоді справедливо, що $len(C2) - len(C1) \geq \tau \cdot s$. Називається τ коефіцієнтом швидкості.

Захоплюється сильне поняття загального префікса. Щодо якості ланцюга, функція μ задовольняє $\mu(a) \geq a$ для протоколів, що цікавлять. В ідеальних умовах μ буде функцією ідентифікації: у цьому випадку відсоток шкідливих блоків у будь-якому досить довгому сегменті ланцюга пропорційний сукупній долі набору (зловмисних) зацікавлених сторін.

Варто зазначити, що для біткойна маємо $\mu(a) = a/(1-a)$, і ця межа насправді є жорсткою, де аргументується така гарантія якості ланцюга. Те ж саме стосується побудови протоколу.

Зростання ланцюга стосується швидкості, з якою росте ланцюг (для чесних сторін). Як і у випадку з біткойнами, *найдовший* ланцюжок відіграє переважну роль у протоколі. Це забезпечує легку гарантію зростання ланцюга.

Модель безпеки. Приймаємо модель для аналізу безпеки протоколів блокчейну, розширеної ідеальною функціональністю F. Позначимо через $VIEW^{P,F}_{\Pi,A,Z}(k)$ вигляд сторони P після виконання протоколу Π з противником A, середовище Z, параметр безпеки k і доступ до ідеальної функціональності F. Зауважимо, що F може охоплювати безліч різних «функціональних можливостей».

Аналіз проводиться в «стандартній моделі» і з безвипадковою функціональністю оракула. Тим не менш, використовується функції «дифузії» та «Ключ-транзакція» з наступними інтерфейсами, описаними нижче.

– *Дифузна функціональність.* Сторона, якщо вона активована, може в будь-який момент отримати вміст свого вхідного рядка, тому це можна вважати поштою. Крім того, сторони можуть дати інструкцію функціональності для розповсюдження повідомлення. Функціональність зберігає раунди (так звані слоти), і всім сторонам дозволяється розпо-

всюджуватися один раз за раунд. Раунди не проходять вперед, якщо всі сторони не розповсюдили повідомлення. Зловмисник, коли його активовано, також може взаємодіяти з функцією, і йому дозволено читати всі «вхідні» папки та всі розповсюджені запити та доставляти повідомлення в папки вхідних у будь-якому порядку. Наприкінці раунду функціональні можливості забезпечують, щоб усі «вхідні» папки містили всі повідомлення, які були розповсюджені (але не обов'язково в тому ж порядку, по якому їх було розповсюджено). Будь-яка сторона може запитати поточний індекс слоту в будь-який час. Якщо зацікавлена сторона не отримує в певному слоті повідомлення, записані в його вхідний рядок слоти очищаються.

– *Функціональність ключів і транзакцій.* Функціональність реєстрації ключа ініціалізується за допомогою n користувачів, $U_1, ..., U_n$ та їх відповідними ставками s_1,...,s_n; враховуючи таку ініціалізацію, функціональність буде консультуватися з супротивником і буде прийнята (можливо порожню) послідовність (Недостовірний, U) повідомлень і позначити відповідних користувачів U як пошкоджених. Для недостовірних користувачів без зареєстрованого відкритого ключа функціональність дозволить супротивнику встановлювати їхні відкриті ключі, а для чесних користувачів функціональність вибирає пари відкритих/секретних ключів і записує їх. Відкриті ключі недостовірних користувачів будуть позначені як такі. Згодом може виконуватися будь-яка послідовність наступних дій: (i) Користувач може попросити отримати свій відкритий та секретний ключ, після чого функція поверне його користувачеві; (ii) Може знадобитися весь каталог відкритих ключів, після чого функція поверне його користувачеві, який запитує; (iii) Новий користувач може бути запрошений на створення повідомленням (Створювання, U, C) із середовища, і в цьому випадку функціональність буде виконуватися за такою ж процедурою, що й раніше: вона консультуватиметься з противником щодо статусу пошкодження U та встановить його відкритий і, можливо, секретний ключ залежно від корупційного статусу; крім того, він зберігатиме C як запропонований початковий стан. Функціональність поверне відкритий ключ назад до середовища після успішного завершення

ції взаємодії; (iv) Середовище може запросити транзакцію від імені певного користувача, надавши шаблон для транзакції (який повинен містити унікальний одноразовий номер) та одержувача. Функціональність відповідним чином коригує ставку кожної зацікавленої сторони; (v) Супротивник може попросити існуючого користувача пошкодити його через повідомлення (Недостовірний, U). Користувач може бути пошкоджений лише після затримки слотів D; зокрема, після реєстрації запиту на пошкодження секретний ключ буде звільнено після того, як слоти D пройдуть відповідно до лічильника раундів, який підтримується в інтерфейсі дифузії.

Виконання протоколу стосується функціональності F, яка включає в себе дві вищезазначені функції, а також, можливо, додаткові функції. Пошкоджена зацікавлена сторона U передасть весь свій стан , з цього моменту супротивник буде активований замість зацікавленої сторони U. За межами обмежень, накладених F, супротивник може корумпувати зацікавлену сторону, лише якщо йому надано дозвіл оточення Z запускає виконання протоколу. Дозвіл надається у формі повідомлення (Недостовірний, U), яке надається противнику оточенням. Підсумовуючи, щодо активації маємо наступне.

- У кожному слоті sl_j середовища Z дозволяється активувати будь-яку підмножину зацікавлених сторін, яку вона бажає. Кожен з них, можливо, створить повідомлення, які будуть передані іншим зацікавленим сторонам.
- Супротивник активується принаймні як останній об'єкт у кожному sl_j, (а також під час усіх активацій супротивної сторони).

Легко помітити, що наведена вище модель надає супротивнику таку широку силу, що неможливо встановити будь-які істотні гарантії щодо протоколів, що представляють інтерес. Таким чином, слід обмежити середовище належним чином (враховуючи деталі протоколу), щоб аргументувати безпеку. Розглянемо обмеження які накладються на середовище.

Обмеження, накладені на навколишнє середовище. Середовище, яке відповідає за активацію чесних сторін у кожному раунді, буде підлягати наступним обмеженням щодо активації чесних сторін, які керують протоко-

лом.

- У кожному слоті буде принаймні одна чесна активована сторона (незалежно від того, чи є вона лідером слота).
- Буде параметр $k \in Z$, який позначатиме максимальну кількість слотів, в яких чесний акціонер може перебувати в автономному режимі. Якщо чесний стейкхолдер(посередник) виникає після початку протоколу за допомогою (Створювання, U, C), його ланцюжок ініціалізації C, наданий середовищем, повинен відповідати ланцюжку чесних сторін, який був активний у попередньому слоті.
- У кожному слоті sl_r і для кожного активного стейкхолдера(посередника) U_j буде набір $S_j(r)$ відкритих ключів і пар ставок виду $(vk_i, s_i) \in 0{,}1^* \cdot N, for_j = 1, ..., n_r$, де n_r – кількість користувачів, представлених до цього слота. Відкриті ключі будуть позначені як «пошкоджені», якщо відповідна зацікавлена сторона була пошкоджена. Будемо говорити, що супротивник обмежений відносною ставкою менше ніж 50%, якщо він вважає, що загальна частка пошкоджених ключів, поділена на загальну частку і $\sum_i s_i$, становить менше 50% у всіх можливих $S_j(r)$. У разі порушення вищезазначеного, подія $Bad^{1/2}$(програш мережи) стає істинною для даного виконання.

Зазначене вище обмеження в автономному режимі є дуже консервативним і протокол може допускати набагато довший час автономного режиму залежно від того, як проходить виконання. Для простоти використовуємо наведене вище обмеження. Зауважимо, що в усіх доказах, коли говориться, що властивість Q має високу ймовірність для всіх виконання, фактично буде стверджуватися, що $Q \vee Bad^{1/2}$ має високу ймовірність для всіх виконань. Це фіксує той факт, що виключаються середовища та супротивники, які запускають $Bad^{1/2}$ з невеликою ймовірністю.

2.2. Огляд протоколу, статичний стан, динамічна ставка

Огляд протоколу

Почнемо з загального огляду підходу до проектування протоколу. Специфіка протоколу залежить від ряду наступних параметрів: (і) k –

кількість блоків, яку певне повідомлення має мати «поверх нього», щоб стати частиною незмінної історії книги; (ii) ϵ – це перевага за ставкою чесних стейкхолдерів перед змагальними; (iii) D – це затримка корупції, яка накладається на супротивника, тобто чесна зацікавлена сторона буде пошкоджена після слотів D, коли супротивник доставить пошкоджене повідомлення під час виконання; (iv) L – термін служби системи, виміряний у слотах; (v) R – довжина епохи, виміряна в прорізах.

Розглядаємо опис протоколу в чотири етапи, послідовно покращуючи модель змагальності, яку він може витримати. На всіх етапах учасникам доступна «ідеальна функціональність» $F_{LS}^{D,F}$. Функціональність фіксує ресурси, які доступні сторонам як передумови для безпечної роботи протоколу.

Етап 1: Статична ставка; $D = L$. На першому етапі припущення довіри є статичним і залишається з початковим набором зацікавлених сторін. Існує початковий розподіл ставок, який жорстко закодований у блок генезису, який включає відкриті ключі зацікавлених сторін, $\{(vk_i, s_i)\}_i^n = 1$. Виходячи з наших обмежень щодо навколишнього середовища, передбачається, що серед цих початкових зацікавлених сторін чесна більшість із перевагою. Середовище спочатку дозволить корумпувати низку зацікавлених сторін, чия відносна частка становить $\frac{1-\epsilon}{2}$ для деяких $\epsilon > 0$. Середовище дозволяє корупцію партій, надаючи супротивникові маркери форми (Недостовірний, U); зауважимо, що через затримку корупції, накладену на цьому першому етапі, будь-яка подальша корупція буде протистояти партіям, які спочатку не мають жодної участі. Отже, модель корупції схожа на «стачну корупцію». $F_{LS}^{D,F}$ згодом зробить вибірку ρ, яка почне «зважену за зацікавленими сторонами» вибірку зацікавлених сторін і таким чином призведе до обрання підмножини з m ключів vk_{i_1},...,vk_{i_m} для формування комітету, який матиме чесну більшість із переважна ймовірність у m (це використовує той факт, що відносна частка, якою володіють зловмисники, становить $\frac{1-\epsilon}{2}$; на цьому етапі буде накладена лінійна залежність від m до ϵ^{-2}). Більш детально, комітет буде обраний неявно шляхом призначення зацікавленої сторони з імовірністю, пропорційною її частці, до кожного з L слотів. Згодом зацікавлені сторони видаватимуть блоки за графіком, який визначається призначенням слотів. Буде застосовано правило найдовшого ланцюга, і супротивник зможе

розділити погляди на блокчейн чесних сторін. Тим не менш, доведемо за допомогою аргументу ланцюга Маркова, що ймовірність того, що вилка може підтримуватися над послідовністю з n слотів, експоненціально падає з принаймні $\sqrt{n}$.

Етап 2: Динамічний стан з маяком, період епохи R слотів, $D = R << L$. Центральна ідея для продовження терміну життя вищезазначеного протоколу полягає в тому, щоб розглянути послідовну композицію кількох його викликів. Детально розповімо, як це зробити, за умови, що надійний маяк випромінює рівномірно випадковий рядок через регулярні інтервали. Точніше, маяк під час проміжків $\{j \cdot R + 1, ..., (j + 1)R\}$ розкриває j-й випадковий рядок, який задає функцію вибору лідера. Критичною відмінністю в порівнянні зі статичним протоколом стану є те, що розподіл ставок може змінюватися і витягується з самого блокчейну. Це означає, що в певному слоті sl, який належить до j-ї епохи (з $j \geq 2$), використовується розподіл ставок, який повідомляється в останньому блоці з міткою часу менше $j \cdot R - 2k$.

Що стосується розподілу часток, що розвиваються, транзакції будуть постійно генеруватися та передаватися між зацікавленими сторонами через середовище, а гравці включатимуть опубліковані транзакції в реєстри на основі блокчейну, які вони ведуть. Для розміщення нових облікових записів, які створюються, функція $F_{LS}^{D,F}$ дозволяє створити новий (vk, sk) за запитом і призначити його новій стороні U_i. Зокрема, середовище може створити нові сторони, які будуть взаємодіяти з $F_{LS}^{D,F}$ для свого відкритого/секретного ключа, таким чином розглядаючи його як надійний компонент, який зберігає таємницю їхнього гаманця. Зауважимо, що супротивник може втрутитися в створення нової сторони, зіпсувати її та надати замість цього власний (створений супротивником) відкритий ключ. Як і раніше, середовище може запитувати транзакції між обліковими записами від зацікавлених сторін, а також може генерувати транзакції у співпраці з супротивником від імені пошкоджених облікових записів. Нагадуємо, що припущення полягає в тому, що в будь-якому слоті, на думку будь-якого чесного гравця, розподіл зацікавлених сторін задовольняє чесну більшість перевагою ϵ (зверніть увагу, що різні чесні гравці можуть сприймати різний розподіл зацікавлених сторін

у певному слоті). Ставка може зміщуватися не більше ніж на σ статистичну відстань протягом певної кількості слотів. Статистична відстань тут буде вимірятися, враховуючи базовий розподіл, який є зваженим за ставками вибірки, і те, як він змінюється протягом зазначеного інтервалу часу. Доказ безпеки можна розглядати як індукцію в кількості епох L/R з базовим випадком, наданим доказом протоколу статичної ставки. Наприкінці будемо стверджувати, що в цьому випадку ставка $\frac{1-\epsilon}{2} - \sigma$, обмежена змагальною ставкою, є достатньою для безпеки одного розіграшу (і зауважимо, що розмір комітету, m, тепер слід вибрати, щоб подолати також адитивний член-розмір $ln(L/R)$, враховуючи, що час життя систем включає таку кількість послідовних епох). Затримка пошкодження залишається на рівні $D = R$, яку можна вибрати довільно меншою за L, що дозволяє противнику виконувати адаптивні пошкодження, поки це не відбувається миттєво.

Етап 3: Динамічний стан без маяка, період епохи R слотів, $R = \Theta(k)$ і затримка $D \in (R,2R) << L$. На третьому етапі усуваємо залежність від маяка, вводячи безпечний багатосторонній протокол із «гарантованою доставкою вихідних даних», який імітує його. Таким чином, можемо отримати довговічність протоколу, як описано в проекті етапу 2, але лише за умови, що дизайн етапу 1, тобто просто наявність початкового випадкового рядка та початкового розподілу зацікавлених сторін із чесною більшістю. Основна ідея полягає в наступному: враховуючи, що гарантується, що чесна більшість серед обраних зацікавлених сторін буде триматися з дуже високою ймовірністю, можемо надалі використовувати цей обраний набір як учасників екземпляра протоколу безпечних багатосторонніх обчислень (MPC). Це вимагатиме вибору довжини епохи, щоб вона могла вмістити протокол MPC. З точки зору безпеки, основна відмінність від попереднього випадку полягає в тому, що вихідний сигнал маяка стане відомим супротивнику до того, як стане відомим чесним сторонам. Тим не менш, доводиться, що чесні сторони також неминуче дізнаються про це після невеликої кількості слотів. Щоб врахувати той факт, що супротивник отримує цю перевагу (яку він може використовувати, виконуючи адаптивні пошкодження), збільшуємо час очікування пошкодження з R до відповідного значення в $(R,2R)$, що зводить даремно цю перевагу і залежить від безпечного MPC. Особливістю

цього етапу з точки зору криптографічного дизайну є використання самого реєстру для моделювання надійної трансляції, яка підтримує протокол MPC.

Етап 4: Підтримувачі, представники зацікавлених сторін, анонімне спілкування. На останньому етапі проектування доповнюємо протокол двома новими ролями для сутностей, які запускають протокол, і враховуємо переваги анонімного спілкування. Підтримувачі введення створюють другий рівень підтвердження транзакцій перед включенням блоку. Цей механізм дозволяє протоколу протистояти відхиленням, таким як егоїстичний видобуток, і дозволяє нам показати, що чесна поведінка є приблизною рівновагою Неша за розумних припущень щодо витрат на виконання протоколу. Зауважимо, що вхідні підтримувачі призначаються слотам так само, як і лідери слотів, а вхідні дані, включені в блоки, прийнятні лише в тому випадку, якщо вони схвалені відповідним підтримувачем введення. Функція делегування дозволяє зацікавленим сторонам передавати участь у комітетах вибраним делегатам, які беруть на себе відповідальність зацікавлених сторін за виконання протоколу (включаючи участь у MPC та видачу блоків). Делегування, природно, породжує «пули ставок», які можуть діяти так само, як пули для майнінгу біткойнів. Включивши рівень анонімного зв'язку можемо усунути вимогу затримки корупції, яка накладається на аналіз. Це робиться за рахунок збільшення вимог до часу онлайн для чесних сторін.

Статичний стан

Почнемо з опису протоколу блокчейну π_{SPoS} у налаштуваннях «статичної ставка», де лідери призначаються слотам блокчейну з імовірністю, пропорційною їхній (фіксованій) початковій ставці, яка буде ефективним розподілом ставок протягом усього виконання. Абстрагуємо процес вибору лідера, розглядаючи його просто як «ідеальну функціональність», яка сумлінно виконує процес випадкового розподілу зацікавлених сторін у слоти.

Навіть за умови ідеального процесу призначення лідера аналіз стандартного правила переваг «найдовшого ланцюга» в наших налаштуваннях Proof-of-Stake, потребує значних нових ідей. Проблема виникає через те, що великі колекції слотів (епох, як описано вище) призначаються зацікавленим сторонам відразу, хоча це має сприятливі властивості з точки зору ефе-

ктивності (і стимулу), воно надає противнику нові засоби атаки. Зокрема, супротивник, який контролює певну групу зацікавлених сторін, може на початку епохи вибрати, коли стандартні трансляційні повідомлення «оновлення ланцюга» будуть доставлені чесним сторонам з повним знанням про майбутні призначення слотів зацікавленим сторонам. Навпаки, противники в типових налаштуваннях Proof-of-Work змушені приймати такі рішення в режимі онлайн.

У випадку статичної ставки припускаємо, що фіксована колекція з n зацікавлених сторін $U_1, ..., U_n$ взаємодіє протягом усього протоколу. Зацікавлена сторона U_i володіє ставкою s_i до початку протоколу. Для кожної зацікавленої сторони U_i генерується пара ключів верифікації та підпису (vk_i, sk_i) для заданої схеми підпису, без втрати загальності припускаємо, що ключі перевірки $vk_1, ..., vk_i$ відомі всім зацікавленим сторонам.

Означення 2.1. Блок генезису B_0 містить список зацікавлених сторін, ідентифікованих їхніми відкритими ключами, їхніми відповідними ставками $(vk_1, s_1), ..., (vk_n, s_n)$ та допоміжною інформацією ρ.

Передбачливо зазначаємо, що допоміжна інформація ρ буде використана для процесу виборів лідера слота.

Означення 2.2. Стан — це рядок $st \in \{0,1\}^\lambda$

Означення 2.3. Блок B, згенерований у слоті $sl_i \in \{sl_1, ..., sl_R\}$, містить поточний стан $st \in \{0,1\}^\lambda$, дані $d \in \{0,1\}^*$, номер слоту sl_i та підпис $\sigma = Sign_{sk_i}(st, d, sl)$, обчислений під sk_i, що відповідає зацікавленій стороні U_i, яка генерує блок.

Означення 2.4. Блокчейн (або просто ланцюжок) відносно блоку генезису B_0 - це послідовність блоків $B_1, ..., B_n$, пов'язаних із строго зростаючою послідовністю слотів, для яких стан st_i для B_i дорівнює $H(B_i - 1)$, де H — задана хеш-функція, стійка до зіткнень. Довжина ланцюга $len(C) = n$ - це кількість блоків. Блок B_n є головою ланцюга, що позначається $head(C)$. Розглядаємо порожній рядок ε як законний ланцюг і за умовою встановлюємо $head(\varepsilon) = \varepsilon$.

Нехай C — ланцюг довжини n, а k — будь-яке ціле невід'ємне число. Позначимо через C^k ланцюжок, що утворюється в результаті видалення k крайніх правих блоків C. Якщо $k \geq len(C)$, визначаємо $C^k = \varepsilon$. Нехай $C_1 \leq C_2$ вказує, що ланцюг C_1 є префіксом ланцюга C_2.

Означення 2.5. Епоха — це набір R суміжних слотів $S = \{sl_1, ..., sl_R\}$.

Означення 2.6. (Коефіцієнт змагальності) Нехай U_A — множина зацікавлених сторін, контрольованих противником A. Тоді коефіцієнт змагальності визначається як

$$\alpha = \frac{\sum_{j \in U_A} s_j}{\sum_{i=1}^{n} s_i}$$

де n – загальна кількість зацікавлених сторін, а s_i – частка зацікавленої сторони U_i.

Вибір лідера слота. У протоколі для кожного $0 < j \leq R$, лідер слоту E_j визначається, хто має (єдине) право генерувати блок на sl_j. Зокрема, для кожного слоту зацікавлена сторона U_i вибирається як лідер слота з ймовірністю p_i, пропорційною його частці, зареєстрованій у блоці генезису B_0, ці призначення є незалежними між слотами. У цьому статичному випадку ставки, блок генезису, а також процедура вибору лідерів слотів визначаються ідеальною функціональністю $F_{LS}^{D,F}$. Ця функція параметризована списком $\{(vk_1, s_1),...,(vk_n, s_n)\}$, що призначає кожній зацікавленій стороні відповідну частку, розподіл D, який надає допоміжну інформацію ρ, і функцію вибору лідера F, визначену нижче.

Означення 2.7. (Процес вибору лідера) Процес вибору лідера щодо розподілу зацікавлених сторін $S = \{(vk_1, s_1),...,(vk_n, s_n)\}$, (D,F) — це пара, що складається з розподільної та детермінованої функції, так що, коли $\rho \leftarrow D$ має значення, що для всіх $sl_j \in sl_1,...,sl_R$, $F(S, \rho, sl_j)$ виводить $U_i \in U_1, ..., U_n$ з імовірністю

$$p_i = \frac{s_i}{\sum_{k=1}^{n} s_k}$$

де s_i – частка, яку має зацікавлена сторона U_i («зважування ставкою»); крім того, сімейство випадкових величин $\{F(S, \rho, sl_j)\}_{j=1}^{R}$ - незалежні.

Зауважимо, що вибірка, пропорційна ставці, може бути реалізована простим способом. Наприклад, простий процес працює таким чином. Нехай $\widetilde{p_i} = s_i / \sum_{j=i}^{n} s_j$, для кожного $i = 1, ..., n-1$, за умови, що жодна зацікавлена сторона ще не вибрана, процес підкидає монету зі зміщенням $\widetilde{p_i}$, якщо результат монети дорівнює 1, сторона U_i вибирається для слота, і процес завершено. (Зверніть увагу, що $\widetilde{p_n} = 1$, тому процес напевно завершиться з унікальним лідером.) Коли реалізується цей процес як функцію $F(\cdot)$, має бути виділено достатню випадковість, щоб імітувати зміщені підкидання монети. Якщо реалізується вищезгадане з точністю λ для кожного окремого підкидання монети, тоді для вибору зацікавленої сторони знадобиться всього $n[log\lambda]$ випадкових бітів. Зауважте, що за допомогою генератора псевдовипадкових чисел (PRG) можна використовувати коротший рядок «насіннєвий» і потім розтягнути його за допомогою PRG до відповідної довжини.

Функціональність $F_{LS}^{D,F}$

$F_{LS}^{D,F}$ включає розповсюдження та ключ/транзакцію з розділу 2 і параметризується відкритими ключами та відповідними ставками початкових зацікавлених сторін $S_0 = \{(vk_1, s_1), ..., (vk_n, s_n)\}$, розподілом D та функцією F, так що (D, F) є процесом вибору лідера. Крім того, $F_{LS}^{D,F}$ взаємодіє із зацікавленими сторонами наступним чином:

– Після отримання (запит головного блок, U_i) від зацікавлених сторін U_i, $F_{LS}^{D,F}$ діє наступним чином. Якщо ρ не встановлено, $F_{LS}^{D,F}$ зразків $\rho \leftarrow D$. У будь-якому випадку, $F_{LS}^{D,F}$ посилає (головний блок, S_0, ρ, F) до U_i

Протокол у $F_{LS}^{D,F}$ – гібридна модель. Опишемо простий протокол блокчейна на основі Proof-of-Stake, враховуючи статичну частку в $F_{LS}^{D,F}$ – гібридній моделі, тобто де блок генезису B_0 (і, отже, лідери слотів) визначаються ідеальною функціональністю $F_{LS}^{D,F}$. Зацікавлені сторони $U_1, ..., U_n$ взаємодіють між собою та з $F_{LS}^{D,F}$ за допомогою протоколу π_{SPoS}.

Протокол спирається на функцію $maxvalid_S(C, C)$, яка вибирає ланцюжок з урахуванням поточного ланцюга C і набору дійсних ланцюжків C, доступних у мережі. У статичному випадку аналізуємо просте правило «найдовшого ланцюга». (У динамічному випадку правило параметризується

загальною довжиною ланцюга)

Функція $maxvalid(C, C)$: повертає найдовший ланцюг із $C \cup C$. Зв'язки розриваються на користь C, якщо вона має максимальну довжину, або довільно в іншому випадку.

Розгалуження рядка

У аргументах безпеки використовуємо елементи $\{0,1\}^n$, щоб вказати, які слоти — серед конкретного вікна слотів довжини n — були призначені учасникам змагань. Коли рядки мають таку інтерпретацію – характерними рядками.

Означення 2.8. (Характеристичний рядок) Виправити виконання з блоком генезису B_0, противником A та середовищем Z. Нехай $S = \{sl_{i+1}, ..., sl_{i+n}\}$ позначає послідовність слотів довжиною $|S| = n$. Характеристичний рядок $w \in \{0,1\}^n$ для S визначається так, що $w_k = 1$ тоді і тільки тоді, коли супротивник контролює лідер слота sl_{i+k}. Для такого характеристичного рядка $w \in \{0,1\}^*$ говоримо, що індекс i є змагальним, якщо $w_i = 1$, і чесним у протилежному випадку.

Починаємо з інтуіції щодо нашого підходу до аналізу протоколу. Нехай $w \in \{0,1\}^n$ — характеристичний рядок для послідовності слотів S. Розглянемо два спостерігачі, які (i) переходять у режим офлайн безпосередньо перед початком S; (ii) мають такий самий вигляд C_0 поточного ланцюга до початку S; (iii) повертаються в режим онлайн на останньому слоті S та запитують оновлення їх ланцюга. Основною проблемою в нашому аналізі є можливість того, що таким спостерігачам може бути представлений «розбіжний» погляд на послідовність S: зокрема, можливість того, що супротивник може змусити двох спостерігачів прийняти два різних ланцюга C_1, C_2, загальний префікс яких є C_0.

Помічаємо, що не всі характерні рядки дозволяють це. Наприклад, (цілком чесний) рядок 0^n гарантує, що два спостерігачі приймуть той самий ланцюжок C, який буде складатися з n нових блоків поверх загального префікса C_0. З іншого боку, інші рядки не гарантують такого спільного розширення C_0, у випадку 1^n, супротивник може створити дві абсолютно

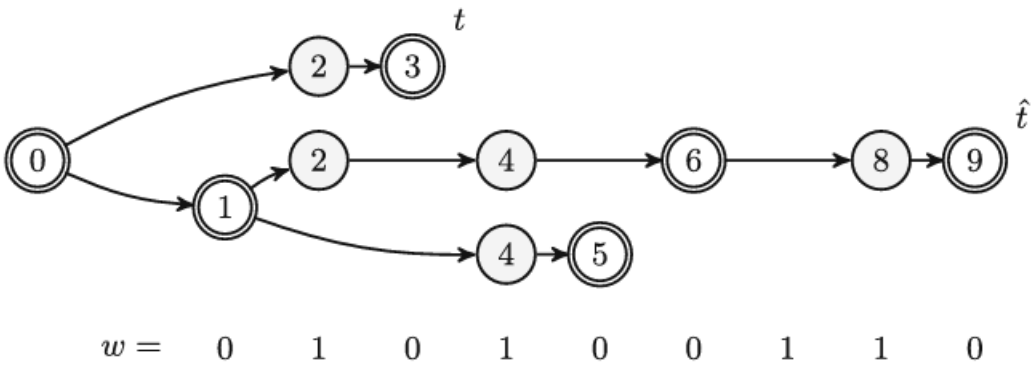
різні історії під час послідовності слотів S і таким чином надати двом спостерігачам два різні ланцюжки C_1, C_2, які мають лише загальний префікс C_0. В решті частини цього розділу встановлюємо, що рядки, які допускають такі «розділи», є досить рідкісними — насправді, показуємо, що вони мають щільність $2^{-\Omega}(\sqrt{n})$ до тих пір, поки частка протиборчих слотів дорівнює $1/2 - \epsilon$.

Щоб міркувати про такі «розгалуження» характеристичного рядка $w \in \{0,1\}^n$, нижче визначається формальне поняття «форк», яке фіксує відносини між ланцюжками, які транслюються чесними лідерами слотів під час виконання протоколу π_{SPoS}. Чесні гравці завжди вирішують розширити ланцюг максимальної довжини серед тих, які доступні гравцеві в мережі. Крім того, якщо такий максимальний ланцюжок C включає блок B, який раніше транслювався чесним гравцем, префікс C перед B повинен повністю узгоджуватися з ланцюжком, який транслював цей попередній чесний гравець. Ця властивість «злиття» безпосередньо випливає з того факту, що стан будь-якого чесного блоку фактично прив'язується до унікального ланцюга, що починається з блоку генезису. На завершення будь-який ланцюжок C, що транслюється чесним гравцем, повинен починатися з ланцюга, створеного раніше чесним гравцем (або, альтернативно, блоком генезису), продовжуватися, можливо, порожньою послідовністю протиборчих блоків і закінчуватися чесним гравцем. Звідси випливає, що ланцюги, які транслюють чесні гравці, утворюють природне спрямоване дерево. Той факт, що чесні гравці надійно транслюють свої ланцюжки і завжди будують на найдовшому доступному ланцюжку, вводить другу важливу властивість цього дерева: «глибини» різних чесних блоків, доданих чесними гравцями під час протоколу, повинні бути різними.

Фактичні ланцюжки, викликані виконанням π_{SPoS}, складаються з блоків, що містять різноманітні дані, які не мають значення для міркування про роздвоєння. Формальне поняття форк відображає орієнтоване дерево, сформоване відповідними ланцюжками та ідентичностями гравців — виражених у вигляді індексів у рядку w — відповідаючі за генерування блоків у цих ланцюгах.

Розгалуження(форк) та розгалуження рядка. Нижче визначаємо

основні комбінаторні структури, які використовуємо, щоб міркувати про можливі погляди, які спостерігаються чесними гравцями під час виконання протоколу з цим характерним рядком.



Форк F для строки $w = 010100110$; вершини відображаються зі своїми мітками, а чесні вершини виділені подвійними межами. Зверніть увагу, що глибини (чесних) вершин, пов'язаних з чесними індексами w, строго зростають. На малюнку позначено два зубці: один, позначений $\hat{t}$, закінчується у вершині, позначеній 9, і є найдовшим зубцем у вилці; другий зубець t закінчується у вершині, позначеній 3. Величина $gap(t)$ вказує різницю в довжині між t і $\hat{t}$;в даному випадку $gap(t) = 4$. Резерв кількості $reserve(t) = |\{i \quad | \quad l(v) < i \leq |w| \text{ та } w_i = 1\}|$ вказує кількість протилежних індексів, що з'являються після мітки останньої чесної вершини v зубця; у цьому випадку $reserve(t) = 3$. Оскільки кожен аркуш F є чесним, F – закритий.

Означення 2.9. (Розгалуження) Нехай $w \in \{0,1\}^n$ і $H = \{i \mid w_i = 0\}$ позначають набір чесних індексів. Розгалуження для рядка w являє собою спрямоване корінне дерево $F = (V, E)$ з позначенням $l : V \to \{0, 1, ..., n\}$, так що

- кожне ребро F спрямоване від кореня;
- корінь $r \in V$ має мітку $l(r) = 0$;
- мітки вздовж будь-якої спрямованої траєкторії в дереві строго збільшуються;
- кожен чесний індекс $i \in H$ є міткою однієї вершини F;
- функція $\mathbf{d} : H \to \{1, ..., n\}$, визначена так, що $\mathbf{d}(i)$ — це глибина у F

унікальної вершини v, для якої $l(v) = i$, строго зростає. (Конкретно, якщо $i, j \in H$ та $i < j$, то $d(i) < d(j)$.)

Для позначення пишемо $F \vdash w$, щоб вказати, що F є розгалуженням для рядка w. Говориться, що розгалуження тривіальне, якщо вона містить одну вершину, корінь.

Означення 2.10. (Зубці і висота) Шлях у розгалуженні F, що походить від кореня, називається зубцем. Для того, щоб довжина (t) позначала його, позначаємо його довжину, що дорівнює кількості ребер на шляху. Висота вилки (як зазвичай для дерева) визначається як довжина найдовшого зубця. Для двох зубців t_1 і t_2 вилки F пишемо $t_1 \sim t_2$, якщо вони мають спільне ребро. Зауважимо, що $\sim$ – відношення еквівалентності на множині нетривіальних зубців; з іншого боку, якщо t означає «порожній» зубець, що складається виключно з кореневої вершини, то $t \nsim t$ для будь-якого зубця t.

Якщо вершина розгалуження v позначена індексом змагальності (тобто $w_{l(v)} = 1$), говориться, що вершина є змагальною; інакше, що вершина є чесною. Для зручності оголошуємо кореневу вершину чесною. Поширюємо цю термінологію на зубці: зубець є чесним, якщо він закінчується чесною вершиною, і змагальним у протилежному випадку. За цією умовою порожній тин є чесним.

Означення 2.11. Кажуть, що розгалуження є плоским, якщо вона має два зубці $t1 \nsim t2$ довжиною, що дорівнює висоті розгалуження. Рядок $w \in \{0,1\}^*$ називається розгалуженим, якщо існує плоске розгалуження $F \vdash w$.

Зауважимо, що для того, щоб виконання π_{SPoS} дало два абсолютно не перетинаних ланцюга максимальної довжини, характерний рядок, пов'язаний із виконанням, повинен бути розгалуженим. Наша мета полягає в тому, щоб встановити наступну верхню межу для кількості розгалужених рядків.

Теорема 2.1. *Нехай $\epsilon \in (0,1)$ і w — рядок, витягнутий з $\{0,1\}^n$ шляхом незалежного призначення кожному $w_i = 1$ з імовірністю $(1-\epsilon)/2$. Тоді $Pr[$ w можна роздвоювати $] = 2^{-\Omega(\sqrt{n})}$.*

Структурні особливості форків: закриті форки, префікси, охоплення та маржа. Починаємо з визначення природного поняття включення для двох розгалужень:

Означення 2.12. (Префікси розгалужень) Якщо w є префіксом рядка $w \in \{0,1\}^*$, $F \vdash w$, та $F' \vdash w'$, говоримо, що F є префіксом F', записується $F \sqsubseteq F'$, якщо F є послідовно позначеним підграфом F'. Зокрема, кожна вершина та ребро F з'являється у F', крім того, мітки, надані будь-якій вершині, що з'являється як у F, так і у F' – ідентичні.

Якщо $F \sqsubseteq F'$, кожен зубець F з'являється як префікс зубця в F'. Зокрема, мітки, що з'являються на будь-якому зубці, що закінчується спільною вершиною, ідентичні, і, крім того, глибина будь-якої чесної вершини, що з'являється як у F, так і в F' – ідентична.

У багатьох випадках зручно працювати з форками, які не «фіксують» нічого за межами кінцевих чесних індексів.

Означення 2.13. (Закриті форки) Форк називається закритим, якщо кожен лист чесний. За умовою тривіальне розгалуження, що складається виключно з кореневої вершини – закрите.

Закрите розгалуження(форк) має унікальний найдовший зубець (оскільки всі максимальні зубці закінчуються чесною вершиною, і вони повинні мати різні глибини). Крім того, якщо w є префіксом w' і $F \vdash w$, то існує унікальна закрите розгалуження $F' \vdash w'$, для кожного $F \sqsubseteq F'$.

Означення 2.14. (Розрив, резерв і досягнення) Нехай $F \vdash w$ — замкнене розгалуження, а $\hat{t}$ позначає унікальний зубець максимальної довжини в F. Ми визначаємо проміжок зубця t, позначений $gap(t)$, як різницю в довжині між $\hat{t}$ і t; таким чином

$$\text{gap}(t) = \text{length}(\hat{t}) \text{ - length}(t)$$

Визначаємо резерв зубця t як кількість змагальних індексів, що з'являються в w після останнього індексу в t; зокрема, якщо t задається шляхом $(r, v_1, ..., v_k)$, де r - корінь з F, визначаємо

$$reserve(t) = |\{i \quad | \quad w_i = 1 \text{ та } i > l(v_k)\}|$$

Зауважимо, що ця величина залежить як від F, так і від конкретного рядка w, пов'язаного з F. Для відрізка t визначаємо

$$\text{reach}(t) = \text{reserve}(t) \text{ - } \text{gap}(t)$$

Означення 2.15. (Маржа) Для закритої вилки $F \vdash w$ визначаємо $\lambda(F)$ як максимальне охоплення всіх зубців у F:

$$\lambda(F) = \max_t \text{reach}(t)$$

Аналогічно, визначаємо межу F, позначену $\mu(F)$, як «передостаннє» осягнення, застосоване для непересічних зубців F: конкретно

$$margin(F) = \mu(F) = \max_{t_1 \nsim t_2}(min\{reach(t_1), reach(t_2)\}) \tag{2.1}$$

Наведені вище максимуми завжди можна отримати чесними зубцями. Зокрема, якщо t є протиборчим зубцем розвилки $F \vdash w$, то $\text{reach}(t) \leq \text{reach}(t)$, де найдовший чесний префікс t.

Оскільки $\sim$ є відношенням еквівалентності на непорожніх зубцях, тоді завжди існує пара зубців t_1 і t_2 (не перетинаються з ребрами), які досягають максимуму у визначальному рівнянні (2.1), які задовольняють $\text{reach}(t_1) = \lambda(F) \geq \text{reach}(t2) = \mu(F)$.

Положення 1 Рядок w можна розгалужувати тоді і тільки тоді, коли існує замкнуте розгалуження $F \vdash w$, для якої $\text{margin}(F) \geq 0$.

Доведення. Якщо w не має чесних індексів, то тривіальне розгалуження, що складається з одного кореневого вузла, є плоским, замкнутим та має невід'ємне поле; отже, обидві умови еквівалентні. Розглянемо розширюваний рядок w з принаймні одним чесним індексом і, нехай, $\hat{i}$ позначає

найбільший чесний індекс w. Нехай F — плоске розгалуження для w. Як згадувалося вище, існує унікальне закрите розгалуження $\overline{F} \vdash w$, яке можна отримати з F шляхом видалення будь-яких протилежних вершин з кінців зубців F. Звернемо увагу, що зубець $\hat{t}$, що містить $\hat{i}$, є найдовшим зубцем у $\overline{F}$, оскільки це найбільший чесний індекс w. З іншого боку, F є плоскою, і в цьому випадку є два зубці t_1 і t_2, що не перетинаються по краях, з довжиною принаймні $\hat{t}$. Префікси цих двох зубців у $\overline{F}$ повинні чітко мати резерв не менше, ніж проміжок (і невід'ємний діапазон); таким чином $\text{margin}(\overline{F}) \geq 0$ за бажанням.

З іншого боку, припустимо, що w має замкнуте розгалуження з $\text{margin}(F) \geq 0$, і в цьому випадку є два непересікаючих ребра зубця F, t_1 і t_2, для яких $\text{reach}(t_i) \geq 0$. Тоді можемо створити плоске розгалуження, просто додавши до кожного t_i шлях вершин $\text{gap}(t_i)$, позначених наступними змагальними індексами, які обіцяє визначення reserve().

У світлі цієї пропозиції для рядка w зосереджуємо нашу увагу на величинах.

$$\lambda(w) = \underset{\substack{F \vdash w \\ F closed}}{max} \lambda(F), \quad \mu(w) = \underset{\substack{F \vdash w \\ F closed}}{max} \mu(F)$$

для зручності,

$$\mathbf{m}(w) = (\lambda(w), \mu(w)).$$

Зауважимо, що це перевантажує позначення $\lambda(\cdot)$ і $\mu(\cdot)$, тому вони застосовуються як до розгалужень, так і до рядків, але налаштування буде зрозумілим з контексту. Визначення апріорі не гарантують, що $\lambda(\cdot)$ і $\mu(\cdot)$ можуть бути досягнуті за допомогою одного і того ж розгалужння(форка). У будь-якому випадку зрозуміло, що $\lambda(w) \geq 0$ і $\lambda(w) \geq \mu(w)$ для всіх рядків w; крім того, згідно з положенням 1, рядок w розщеплюється тоді і тільки тоді, коли $\mu(w) \geq 0$. $\mu(w)$ називаємо маржою рядка w.

Маючи ці визначення, розглядаємо доказ теореми 2.1.

Доведення теореми 2.1, огляд високого рівня. Доведення проводиться шляхом встановлення рекурсивного опису $\mathbf{m}(w0)$ і $\mathbf{m}(w1)$ у термінах $\mathbf{m}(w)$ та надання аналізу ланцюга Маркова, який виникає при розгляді $\mathbf{m}(\cdot)$ для рядків, отриманих з біноміального розподілу. Це дає верхню межу

ймовірності того, що $\mu(w) \geq 0$, і подія w є роздвоєною. Повний доказ з'являється в статті [3].

Приховані противники. При спробі супротивник передачі двох різних блоків для певного слота, він залишає підозрілий «аудиторський слід» (кілька підписаних блоків для одного слота), який помітно відхиляється від протоколу. Для збереження фасаду чесності, супротивникам це може бути небажано. Таких супротивників називають «прихованими» та вони мають зменшені можливості для порушення протоколу.

Загальний префікс

Ланцюжки, побудовані чесними гравцями під час виконання π_{SPoS}, відповідають зубцям розгалуження. Випадкове призначення слотів зацікавленим сторонам, задане $F_{LS}^{D,F}$, гарантує, що координати асоційованого рядка характеристик w слідують біноміальному розподілу з імовірністю, рівною змагальній ставці. Таким чином, теорема 2.1 встановлює, що жодне виконання протоколу π_{SPoS} не може викликати два зубці (ланцюги) максимальної довжини без спільного префікса.

Однак у контексті π_{SPoS} встановлюємо набагато сильнішу загальну властивість префікса: ланцюжки, про які повідомляють будь-які два чесних гравця, повинні мати «недавній» загальний префікс, у тому сенсі, що видалення невеликої кількості блоків із коротшого ланцюга в результаті утворюється префікс довшого ланцюга.

Теорема 2.2. *Нехай $k, R \in N$ і $\epsilon \in (0,1)$. Імовірність того, що протокол π_{SPoS}, коли він виконується з часткою $a(1-\epsilon)/2$, порушує загальну властивість префікса з параметром k протягом епохи R слотів, становить не більше ніж $exp(-\Omega(\sqrt{k}) + lnR)$; константа, прихована записом $\Omega()$, залежить лише від ϵ.*

Доведення. Повне доведення (див. [3]) продовжується, показуючи, що якщо загальний префікс з параметром k порушується для конкретного розгалуження, тоді базовий рядок характеристик повинен мати підрядок, який можна розгалужувати, довжини k. Таким чином

Pr[поширене порушення префікса] $\leq$

$$\leq Pr\begin{cases}\exists\ \alpha,\beta \in \{1,...,R\} \text{ так, що } \alpha+k-1 \leq \beta \text{ і} \\ w_\alpha ... w_\beta \text{ можуть бути розгалужені}\end{cases} \leq$$

$$\underbrace{\leq \sum_{1\leq R} \sum_{\alpha+k-1\leq\beta\leq R} Pr[w_\alpha ... w_\beta \text{можна розгалужити}]}_{(*)}$$

Характеристичний рядок $w \in \{0,1\}^R$ для такого виконання π_{SPoS} визначається шляхом присвоєння кожному $w_i = 1$ незалежно з ймовірністю $(1-\epsilon)/2$. Відповідно до теореми 2.1, ймовірність того, що рядок довжини t, витягнутий з цього розподілу, є розгалуженим, не більше ніж $exp(-c\sqrt{t})$ для додатної константи c. Зауважимо, що для будь-якого $\alpha \geq 1$,

$$\sum_{t=\alpha+k-1}^{R} e^{-c\sqrt{t}} \leq \int_{k-1}^{\infty} dt = (2/c^2)(1+c\sqrt{k-1})e^{-c\sqrt{k-1}} = e^{-\Omega(\sqrt{k})}$$

і випливає, що сума 2.2 вище $exp(-\Omega(\sqrt{t}))$. Таким чином

Pr[поширене порушення префікса] $\leq R \cdot exp(-\Omega(\sqrt{k})) \leq exp(lnR - \Omega(\sqrt{k}))$

за бажанням.

Зростання ланцюга та якість ланцюга

Передбачаючи ці два докази, записуємо адитивну межу Чернова–Хефдінга. (Див. [4].)

Теорема 2.3. *Нехай $X_1, ..., X_T$ — незалежні випадкові величини з $E[X_i] = p_i]$ і $X_i \in$ [0,1]. Нехай $X = \sum_{i=1}^{T} X_i$ і $\mu = \sum_{i=1}^{T} p_i = E[X]$. Тоді, $\forall \delta \geq 0$,*

$$Pr[X \geq (1+\delta)\mu] \geq e^{-\frac{\delta^2}{2+\delta}\mu} \text{ та } Pr[X \geq (1-\delta)\mu] \leq e^{-\frac{\delta^2}{2+\delta}\mu}$$

Почнемо з властивості росту ланцюга.

Теорема 2.4. *Протокол π_{SPoS} задовольняє властивість росту ланцюга з параметрами $\tau = 1-\alpha$, $s \in N$ протягом епохи R слотів з імовірністю щонайменше $1 - exp(-\Omega(\epsilon^2 s) + lnR)$ проти противника, який має $\alpha - \epsilon$ – частину загальної ставка.*

Доведення. Доведення протікає, застосовуючи обмеженяя Чернофф, який повинен забезпечити високу ймовірність характерного рядка витягнутого з біноміального розподілу, має $a \equiv \tau = (1 - \alpha)$ фракцію чесних показників. Зауважимо, що кожний чесний гравець змусить довжину отриманого ланцюга збільшуватися на будь-яке виконання π_{SPoS}. Див. [3] для повної презентації.

Встановивши ріст ланцюга, розглянемо на якість ланцюга. Властивість якості ланцюга з параметрами μ та l стверджує, що серед усіх послідовних блоків ланцюга (що володіє чесним користувачем), частка змагальних блоків становить не більше μ.

Теорема 2.5. *Нехай $\alpha - \epsilon$ – коефіцієнт змагань. Протокол π_{SPoS} задовольняє властивість якості ланцюга з параметрами $\mu(\alpha - \epsilon) = \alpha/(1 - \alpha)$ та $l \in N$ по всій епоху R слотів принаймні з ймовірністю принаймні з ймовірністю*

$$1 - exp(-\Omega(\epsilon^2 \alpha l) + lnR)$$

Доведення. Це також випливає з відповідного застосування обмеження Чернoффа. Див. [3] для повного обговорення.

Динамічна ставка

Імітація довіреного маяка

Протокол π_{DPoS} обробляє кілька епох і враховує зміни розподілу пакетів, все ще покладаючись на $F_{LS}^{D,F}$ для виконання процесу відбору лідера. Реалізація $F_{LS}^{D,F}$ здійснюється за допомогою протоколу π_{DLS}, що дозволяє зацікавленим сторонам обчислити випадковість та допоміжну інформацію, необхідну на виборах лідера.

Різницею між $F_{LS}^{D,F}$ і $F_{DLS}^{D,F}$ є безперервна генерація випадкових рядків $\rho^2, \rho^3, ...$ для епох $e_2, e_3, ...$ Протокол π_{DLS} використовуватиме протокол кидання монет для генерації неупередженої випадковості, яка може бути використана для визначення значень ρ^j,$j \geq 2$ завантаження на початковий випадковий рядок та початковий розподіл зацікавлених сторін. Невдача протоколу може бути спричинена шляхом перерви викидання монети супротивником. Побудуємо схему кидання монет із "гарантованою доставкою виходу".

Припущення, яке буде використовуватися щодо схеми PVSS, полягає в тому, що отриманий протокол підкидання монет імітує ідеальний маяк з відмінною перевагою ϵ_{DLS}. Симуляція тут припускає, що у випадку чесної більшості існує симулятор, який взаємодіє з супротивником і створює нерозрізнені протокольні стенограми, коли надається значення маяка після етапу зобов'язань. Використовуючи [5] як PVSS, симулятор може досягти симулюваності в моделі випадкового оракула, використовуючи переваги програмованості оракула. У цьому випадку ми не повинні обов'язково використовувати випадковий оракул, ми можемо отримати це з CRS, вбудованого у блок генезису.

Надійна книга транзакцій

Формулюємо основний результат розділу, який встановлює, що протокол π_{DPOS} з протоколом π_{DLS} як підпрограмою реалізує надійний реєстр транзакцій за умов навколишнього середовища. У випадку з динамічною ставкою протрібно переконатися, що супротивник не зможе використати те, як ставка змінюється з часом, і розмістить набір зацікавлених сторін, що

дозволить контролювати більшість виборного комітету зацікавлених сторін у певну епоху. Щоб охопити цю залежність від «зсувів», вводиться наступна властивість.

Означення 2.16. Розглянемо два слоти sl_1, sl_2 та виконання $\mathcal{E}$. Зсув ставки між sl_1, sl_2 – це максимально можлива статистична відстань двох зважених розподілів за ставками, які визначаються за допомогою ставки, відображеної в ланцюжку C_1 деякої чесної зацікавленої сторони, активної на sl_1 та ланцюг C_2 певної чесної зацікавленої сторони, активної на sl_2 відповідно.

Враховуючи наведене вище визначення, тепер можна сформулювати наступну теорему.

Теорема 2.6. *Фіксуємо параметри k,R,$L \in N$, ϵ,$\sigma \in (0,1)$. Нехай $R = 10k$ — довжина епохи, а L – загальний час життя системи. Припустимо, що противник обмежений відносною ставкою $1 - 2 - \sigma$ і що протокол π_{SPOS} задовольняє властивість загального префікса з параметрами R, k і ймовірність помилки ϵ_{CP}, властивість якості ланцюга з параметрами $\mu \geq 1/k$, k і ймовірність помилки ϵ_{CQ} і властивість росту ланцюга з параметрами $\tau \geq 1/2$, імовірністю помилки ϵ_{CG}. Крім того, припустимо, що π_{DLS} моделює ідеальний маяк з відмінною перевагою ϵ_{DLS}.*

Тоді протокол π_{DPOS} задовольняє стабільність з параметрами k і живість з параметрами $u = 2k$ протягом періоду L слотів з імовірністю $1-(L/R)(\epsilon_{CQ}+\epsilon_{CP}+\epsilon_{CG}+\epsilon_{DLS})$, припускаючи що σ – максимальний зсув ставки на 10 тис. слотів, затримка корупції $D \geq 2R$ – 4 тис. і жоден чесний гравець не перебуває в режимі офлайн більше ніж k слотів.

Доведення. Розглянемо виконання π_{DPOS}, коли $F_{LS}^{D,F}$ використовується замість π_{DLS}. Нехай BAD_r – це подія, коли будь-яка з трьох властивостей CP, CQ, CG порушена в раунді $r \geq 1$, при цьому жодного порушення жодного з них не було раніше до r. Легко побачити, що $\Pr[U_{r \leq R} BAD_r] \leq CQ + CP + CG$. Обумовлюючи заперечення цієї події, можна повторити аргумент для другої епохи, оскільки $D \geq R$ і, таким чином, супротивник не може впливати на вибір зацікавленої сторони для другої епохи. Звідси випливає, що $\Pr[U_{r \leq L} BAD_r] \leq (L/R)(\epsilon_{CQ} + \epsilon_{CP} + \epsilon_{CG})$. Тепер легко помітити, що стійкість і життєдіяльність обумовлюють заперечення вищевказаної

події, порушення стійкості порушило б загальний префікс. З іншого боку, порушення живучості порушить або зростання ланцюга, або якість ланцюга для заявлених параметрів.

Результат буде зберігатися, навіть якщо $F_{LS}^{D,F}$ було ослаблено, щоб дозволити противнику отримати доступ до випадкового значення наступної епохи на $6k$ слотів до кінця епохи. Це пояснюється тим, що затримка пошкодження $D \geq 2R - 4k = 16k$.

Дослідимо, що відбувається, коли $F_{LS}^{D,F}$ замінюється на $F_{LS}^{D,F}$ і виконується протокол π_{DLS}. Розглянемо виконання з середовищем Z та противником A та подією BAD, яка відбувається з деякою ймовірністю β у цьому виконанні. Створімо супротивника $A*$, який діє у виконанні з ослабленим $F_{LS}^{D,F}$, як у попередньому абзаці, і викликаємо подію BAD приблизно з тією ж ймовірністю β. $A*$ буде працювати наступним чином: у перших $4k$ слотах він використовуватиме чесну сторону для вставки в блокчейн змодельованих зобов'язань чесних сторін; це можливо для $A*$, оскільки в $4k$ слотах зростання ланцюга призведе до зростання блокчейна щонайменше на $2k$ блоків, і у перші k блоків буде включений принаймні один чесний блок. Тепер $A*$ отримає від $F_{LS}^{D,F}$ значення маяка і буде моделювати відкриття всіх зобов'язань від імені чесних сторін. В останніх $2k$ слотів він виконає примусове відкриття всіх змагальних зобов'язань, які не були відкриті. Моделювання протоколу буде повторюватися для кожної епохи. □

Примітка 2. Модель змагальності легко розширити, включивши в ній корупцію, що зупиняє (і відновлює) відповідно до візантійської корупції. Перевага цієї змішаної корупційної установки полягає в тому, що можна довести, що можна терпіти велику кількість корупційних порушень (свавільно понад 50%). Інтуїція, що стоїть за цим, проста: аналіз розгалужених ланцюгів, все ще застосовується, навіть якщо довільний відсоток лідерів слотів стає неактивним. Єдиним необхідним положенням для цього було б розширення параметра k, обернено пропорційно коефіцієнту незупинених сторін.

2.3. Стимулювання

Аналіз був зосереджений на налаштуванні супротивника з боку криптографії. Далі розглядається створення коаліції раціональних гравців та їх стимули до відхилення від чесної протокольної роботи.

Вхідні прихильники. Для призначення двох різних ролей зацікавленим сторонам потрібно змінити основний протокол для вирішення питання стимулів. Кожна епоха має набір обраних зацікавлених сторін, які керують безпечним багатопартійним протоколом підкидання монет і є лідерами епохи. Разом з ними існує група зацікавлених сторін, яку називають прихильниками. Кожен слот має два типи зацікавлених сторін, пов'язаних з ним: лідер слота, який видасть блок і прихильник слоту, який підтвердить вхід, якого буде включено в блок. Більше того, на відміну від лідерів слотів, можна обрати кілька прихильників слотів для кожного слота, проте, без втрати загальності, просто припускаємо, що в цьому описі є один прихильник для кожного слота. Хоча це здається незначною модифікацією, вона дає простір для вдосконалення з наступної причини: внески прихильників будуть прийнятними, навіть якщо вони запізнилися на d слотів, де $d \in N$ – параметр.

Можна легко помітити, що вдосконалений протокол π_{DPOSwE} має таку ж стійкість і життєздатність, що й π_{DPOS}: модифікація з прихильниками не надає супротивнику жодної можливості перешкодити ланцюжку зростати, приймати вхідні дані або бути послідовним. Однак, якщо оцінити якість ланцюга з точки зору кількості включених схвалених вхідних даних, це дає більш сприятливий результат: легко побачити, що кількість схвалених вхідних даних, що походять від набору зацікавлених сторін S в будь-якій частині ланцюга довжиною k, дорівнює пропорційна відносній ставці S з високою ймовірністю. Це випливає з того факту, що достатньо створити єдиний чесний блок, щоб усі схвалені входи останніх d слотів були включені в нього. Припускаючи $d \geq 2k$, будь-який набір зацікавлених сторін S буде прихильником у підмножині слотів d з імовірністю, пропорційною її кумулятивній ставці, і, таким чином, буде результат.

Відповідний клас механізмів винагород. Механізм винагороди, який

об'єднаємо з прихильниками введення, працює таким чином. Спочатку встановлюємо вікно прийняття схвалення $d = 2k$. Нехай C — ланцюг, що складається з блоків $B_0, B_1, ...$ Розглянемо послідовність блоків, що охоплюють j-ту епоху, позначену $B_1, ..., B_j$ з мітками часу в $\{jR{+}1, ..., (j{+}1)R{+}2k\}$, які містять послідовність $r \geq 0$ затверджених вхідних даних, які походять з j-ї епохи (деякі з них можуть бути включені як частина $j + 1$ епохи). Визначаємо загальний P_R пул винагород, що дорівнює сумі комісій за транзакції, які включені в схвалені вхідні дані, що відповідають j-й епосі. Якщо транзакція відбувається кілька разів (як частина різних схвалених вхідних даних) або навіть у конфліктних версіях, у розрахунку P враховується лише перша поява операції (і вважається частиною книги в цій позиції), де загальний використовуваний порядок зазначається порядком входів прихильників, які включені в C. У послідовності цих блоків визначаємо лідерів слотів за допомогою $L_1, ..., L_R$, що відповідають слотам епохи, і за допомогою $E_1, ..., E_r$ прихильників введення, які внесли послідовність r схвалених введених даних. Згодом i-та зацікавлена сторона U_i може претендувати на винагороду до суми $(\beta \cdot |\{j \mid U_i = E_j\}|/r + (1 - \beta) \cdot |\{j \mid U_i = L_j\}|/R)P$, де $\beta \in [0{,}1]$. Отримання винагороди здійснюється шляхом здійснення транзакції типу «coinbase» в будь-який момент після $4k$ блоків у наступну епоху до тієї, з якої вимагається винагорода.

Зауважимо, що вищезгаданий механізм винагороди має такі особливості: (i) він винагороджує обраних членів комітету за те, що вони просто члени комітету, незалежно від того, випустили вони блок чи ні; (ii) він винагороджує тих, хто підтримує внесок, тими внесками, які вони внесли; (iii) він винагороджує сутності за епоху j після слоту $jR + 4k$.

Продовжуємо показувати, що ця система є рівновагою δ - Неша (наближеною), див. [6]. Зокрема, теорема стверджує, що будь-яка коаліція, яка відхиляється від протоколу, може додати щонайбільше адитивну δ до своєї загальної винагороди.

Технічна складність у наведеному вище формулюванні полягає в тому, що кількість гравців, їх відносна ставка, а також винагороди, які вони отримують, базуються на транзакціях, які генеруються під час виконання самого протоколу. Щоб спростити аналіз, розглянемо налаштування, де

кількість гравців є статичною, ставка, яку вони мають, не змінюється з часом, а вартість виконання протоколу є незначною. Помічаємо, що загальна винагорода (а, отже, і корисність за нашим припущенням щодо витрат на протокол), яку будь-яка коаліція V чесних гравців може отримати від виконання тривалістю $L = tR + 4k + 1$ слотів, дорівнює

$$\mathcal{R}_V(\varepsilon) = \sum_{j=1}^{t} P_{all}^{(j)} \left(\beta \frac{IE_V^j(\varepsilon)}{R} + (1-\beta) \frac{SL_V^j(\varepsilon)}{r_j} \right)$$

для будь-якого виконання ε, де виконується загальний префікс з параметром k, де r_j – загальна кількість затверджених вхідних даних, випущених в j-ту епоху (і, можливо, включені в будь-який час до перших $2k$ слотів епохи $j+1$), $P_{all}^{(j)}$ – це пул винагород епохи j, $SL_V^j(\varepsilon)$ – кількість разів, коли члена V обирали лідером епохи j, а $IE_V^j(\varepsilon)$ – кількість разів, коли члена V було обрано для підтвердити введення в епоху j.

Звернемо увагу, що фактичні винагороди, отримані набором раціональних гравців V під час виконання ε, можуть відрізнятися від $R_V(\varepsilon)$; наприклад, коаліція V може ніколи не схвалити набір вхідних даних, у цьому випадку вони отримають меншу кількість винагород. Крім того, зауважте, що залишаємо значення $R_V(\varepsilon)$ невизначеним, коли ε є виконанням, де звичайний префікс не працює: не має сенсу розглядати це значення для таких виконання, оскільки погляди на протокол чесних сторін можуть відрізнятися; проте це не вплине на наш загальний аналіз, оскільки такі страти відбуватимуться з достатньо малою ймовірністю.

Встановимо той факт, що протокол є рівновагою δ - Неша, довівши, що коаліція V, навіть відхиляючись від належної поведінки протоколу, не може отримати корисність, яка перевищує $R_V(\varepsilon) + \delta$ для деякої відповідної константи $\delta > 0$.

Теорема 2.7. *Зафіксуємо будь-яке $\delta > 0$; чесною стратегією в протоколі є рівновага δ - Неша проти будь-якої коаліції, що має частку ставки менше $(1-\epsilon)/2 - \sigma$ для деяких констант ϵ, $\sigma \in (0,1)$, як у теоремі 2.6, за умови, що максимальна Загальна винагорода P_{all}, що надається в усіх можливих виконанняx протоколів, обмежена поліномом у λ, тоді як*

$\epsilon_{CQ} + \epsilon_{CP} + \epsilon_{CG} + \epsilon_{DLS}$ *є незначним у* λ.

Для доказу дивитись на повну версію статті [3].

Примітка 3. У наведеній вище теоремі для простоти припущено, що витрати на протокол не впливають на кінцеву корисність (по суті, це означає, що витрати на протокол вважаються незначними). Тим не менш, легко розширити доказ, щоб охопити налаштування, де від'ємний член вводиться у функцію виплати для кожного гравця пропорційно кількості підтверджень введених даних і кількості повідомлень, переданих для протоколу MPC. Доказ буде стійким до цих модифікацій, оскільки схвалені введення та повідомлення протоколу MPC не можуть бути задушені супротивником, а отже, функція винагороди може бути розроблена з відповідними ваговими показниками для таких дій, які компенсують їх вартість. Однак зауважте, що надані винагороди вважаються «рівними» як для слотів, так і для схвалених введених даних, і, отже, витрати також повинні бути однаковими. Залишаємо для майбутньої роботи дослідження більш витонченого налаштування, де витрати та винагороди пропорційні фактичним обчислювальним крокам, необхідним для перевірки транзакцій і блоків випуску.

2.4. Делегація долі

У цьому розділі розглянемо схему делегування, за допомогою якої зацікавлені сторони протоколу PoS можуть делегувати права на виконання протоколу іншій групі сторін, делегатам. Делегат може брати участь у протоколі, лише якщо він представляє певну кількість зацікавлених сторін, чия сукупна частка перевищує заданий поріг. Такий поріг участі гарантує, що атака «фрагментації», яка має на меті збільшити кількість делегатів, щоб пошкодити роботу протоколу, не може спричинити великий штраф, оскільки вона здатна змусити розмір комітету, який виконує протокол, бути невеликим (варто зазначити, що механізм делегування подібний до пулів майнінгу в протоколах блокчейну Proof-of-Work).

Схема делегування. Концепція делегування проста: будь-яка зацікавлена сторона може дозволити делегату генерувати блоки від свого імені. У контексті нашого протоколу, коли лідер слоту підписує блок, який він

генерує для певного слота, така схема може бути реалізована простим способом на основі підписів проксі [7].

Зацікавлена сторона може передати право на створення блоків, створивши ключ підпису проксі, який дозволяє делегату підписувати повідомлення у формі (st, d, sl_j) (тобто формат повідомлень, підписаних у протоколі π_{DPoS} для аутентифікації блоку). Щоб обмежити потужність генерації блоків делегата певним діапазоном епох/слотів, зацікавлена сторона може обмежити дійсний простір повідомлень ключа підпису проксі-сервера рядками, що закінчуються номером слота sl_j у певному діапазоні значень. Делегат може використовувати ключ підпису проксі від певної зацікавленої сторони, щоб просто запустити протокол π_{DPoS} від її імені, підписуючи блоки, які ця зацікавлена сторона обрала для створення за допомогою ключа підпису проксі. Ця проста схема є безпечною завдяки властивостям схем підпису проксі-сервера щодо перевірки та запобігання неправомірному використанню, які гарантують, що будь-яка зацікавлена сторона може перевірити, що ключ підпису проксі-сервера був насправді виданий певною зацікавленою стороною певному делегату, і що делегат може використовувати лише ці ключі для підписання повідомлень у межах дійсного простору повідомлень ключа, відповідно. Зауважимо, що хоча підписи проксі-сервера можна описати як загальний примітив високого рівня, такі схеми легко побудувати зі стандартних схем цифрового підпису за допомогою делегування через проксі, як показано в [7]. У цій конструкції зацікавлена сторона підписує сертифікат, що вказує ідентичність делегатів (наприклад, його відкритий ключ) і дійсний простір повідомлень. Пізніше делегат може підписувати повідомлення в межах дійсного простору повідомлень, надаючи підписи для цих повідомлень у рамках з власним відкритим ключем разом із підписаним сертифікатом. Як додаткову перевагу, схеми підпису проксі-сервера також можуть бути побудовані з сукупних підписів таким чином, щоб підписи, згенеровані під ключем підпису проксі-сервера, мали по суті той самий розмір, що й звичайні підписи [7].

Важливим фактором у наведених вище умовах є той факт, що зацікавлена сторона може захотіти відкликати свою підтримку зацікавленій стороні до закінчення терміну дії ключа підписання довіреної особи. Зверніть увагу, що ключі підпису проксі-сервера можна однозначно ідентифікувати. Таким

чином, вони можуть бути відкликані списком відкликаних сертифікатів у блокчейні.

Поріг прийнятності. Делегування, як описано вище, може покращити фрагментацію, яка може виникнути в розподілі ставок. Тим не менш, це не заважає зловмисній зацікавленій стороні розділити свою частку на кілька облікових записів і, утримуючись від делегування, спричинити дуже великий розмір комітету. Для вирішення цього, як згадувалося вище, можна застосувати поріг T, скажімо, 1%. Це означає, що будь-який делегат, який представляє меншу частку, меншу за T від загальної ставки, автоматично не може бути членом комітету. Цьому можна сприяти перерозподіл прав голосу делегатів, які представляють менше T, на інших делегатів детермінованим способом (наприклад, починаючи з тих, хто має найвищу ставку, і розриваючи зв'язки відповідно до лексикографічного порядку). Припустимо, що сформовано комітет $C_1, ..., C_m$ із загальної кількості k розіграшів зважування по ставках. Кожен член комітету проведе k_i таких голосів, де $\sum_{i=1}^{m} k_i = k$. Виходячи з наведеного вище порогу прийнятності, випливає, що $m \leq T^{-1}$ (максимальне значення — це випадок, коли вся ставка розподілена на T^{-1}, делегує кожен власник T ставки).

ВИСНОВКИ

В даній роботі ми вивчали консенсуси Proof-of-Work та Proof-of-Stake, встановили проблеми кожного з цих консесусів, розглянули різні приклади атак супротивників для втручення у ці системи, показали застосування у сучасному світі на зміну банківським структурам. Обидва консенсуси мають певні проблеми у своєму використанні з якими ми ознайомились у роботі. Отже, Proof-of-Work це консенсус, який спирається на доказ роботи і потребує використання великої кількості енергії, у той час як Proof-of-Stake спирається на доказ долі, якою володіє учасник мережи, і не потребую такої кількості енергії як Proof-of-Work.

СПИСОК ЛІТЕРАТУРИ

1. Bitcoin: A Peer-to-Peer Electronic Cash System
2. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol
3. Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: a provably secure proof-of-stake blockchain protocol. Cryptology ePrint Archive, Report 2016/889 (2017)
4. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, New York (1995)
5. Schoenmakers, B.: A simple publicly verifiable secret sharing scheme and its application to electronic voting. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 148–164. Springer, Heidelberg (1999).
6. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.V.: Algorithmic Game Theory. Cambridge University Press, New York (2007)
7. Boldyreva, A., Palacio, A., Warinschi, B.: Secure proxy signature schemes for delegation of signing rights. J. Cryptol. 25(1), 57–115 (2012)