Лабораторна робота №2.

Тема: Реалізація смарт-контракту або анонімної криптовалюти.

Мета роботи: Отримання навичок роботи із смарт-контрактами або анонімними криптовалютами.

Виконали: студенти групи ФІ-32мн Ємець Єлизавета, Карловський Володимир, Коваленко Дар'я

Завдання на лабораторну роботу

1. дослідження методів анонімізації/деанонімізації запропонованої криптовалюти із аналізом складності проведення атак деанонімізації і втрат ефективності анонімних криптовалют у порівнянні із Bitcoin/Litecoin;

2. оцінка та обґрунтування необхідних ресурсів (гасу і ефіру), потрібних для функціонування смарт-контракту.

I. Для дослідження методів анонімізації та деанонімізації пропонованої криптовалюти пройдемося по таких кроках:

1.1. Методи анонімізації

Детальніше розглянемо різні методи анонімізації, які використовуються в криптовалютах, для забезпечення конфіденційності транзакцій та ідентичності користувачів.

1.Міксування Транзакцій (Coin Mixing or CoinJoin)

Механізм: Міксування транзакцій полягає у комбінуванні декількох транзакцій від різних користувачів в одну велику транзакцію з багатьма виходами. Це ускладнює визначення, які вхідні кошти відповідають яким виходам.

Приклади криптовалют: Bitcoin (за допомогою сторонніх сервісів міксування), Dash (функція PrivateSend).

Переваги	**Недоліки**
Зниження ймовірності трасування транзакцій.	Залежність від обсягу міксованих транзакцій (більше учасників — вища анонімність).
Підвищення анонімності без впровадження додаткових криптографічних методів.	Потенційні ризики безпеки при використанні сторонніх сервісів.

2. Zero-Knowledge Proofs (Докази з нульовим розголошенням)

Механізм: Криптографічний метод, що дозволяє одній стороні (пруверу) довести іншій стороні (верифікатору), що даний вислів істинний, не розкриваючи жодної іншої інформації, крім самої істинності вислову.

Приклади криптовалют: Zcash використовує zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) для забезпечення анонімності.

Переваги	**Недоліки**
Високий рівень конфіденційності транзакцій.	Висока обчислювальна складність.
Верифікація без необхідності розкриття додаткової інформації.	Потребує додаткових ресурсів для реалізації та підтримки.

3. Анонімні криптовалютні протоколи

Механізм: Використання спеціалізованих протоколів, які внатурі забезпечують анонімність транзакцій.

Приклади криптовалют:
Monero: використовує Ring Signatures та Stealth Addresses для замаскування відправників і одержувачів.
Verge: застосовує технологію Tor і I2P для приховування IP-адрес користувачів.

Переваги	**Недоліки**
Вбудовані механізми конфіденційності.	Можуть бути проблеми з масштабуванням.
Складніше відслідковувати у порівнянні з Bitcoin.	Часто підозрюються у використанні для нелегальних цілей через високий рівень анонімності.

Ці методи забезпечують різні рівні анонімності та безпеки, залежно від використовуваних технологій і принципів. Кожен метод має свої переваги та недоліки, які впливають на вибір криптовалюти в залежності від потреб користувачів.

1.2. Методи деанонімізації

Детально розглянемо методи деанонімізації у криптовалютах, аналізуючи потенційні вразливості та атаки.

1. Аналіз Трафіку Мережі (Network Traffic Analysis)

Механізм: Атака базується на моніторингу і аналізі трафіку між користувачами мережі, щоб визначити, які вузли відправляють або отримують дані.

Підходи	Заходи протидії
Аналіз кореляції трафіку: Атакуючий може аналізувати час і об'єм трафіку, щоб визначити відносини між вузлами.	Використання мережевих протоколів анонімності, таких як Tor або I2P.
Ендпойнт моніторинг: Шпигування за кінцевими точками з'єднань для виявлення IP-адрес і відповідних транзакцій.	Шифрування всього трафіку для запобігання аналізу.

2. Статистичний Аналіз Розподілу Транзакцій (Transaction Graph Analysis)

Механізм: Вивчення графу транзакцій, щоб виявити шаблони, що можуть вказувати на пов'язані валютні потоки або групи користувачів.

Підходи	Заходи протидії
Кластеризація: Групування адрес на основі частоти і характеру транзакцій між ними.	Використання міксерів або сервісів CoinJoin для змішування транзакцій.
Heuristic Analysis: Використання евристик, наприклад, "одна адреса за виходом" (one-address-per-output).	Розподіл транзакцій між численними адресами (address reuse).

3. Атаки Sybil

Механізм: Створення великої кількості псевдо-вузлів у мережі, щоб оточити цільовий вузол і контролювати його зв’язки.

Підходи	Заходи протидії
Контроль зв'язків: Атакуючий може спробувати стати єдиним шлюзом для зв'язків цілі.	Використання алгоритмів консенсусу, які вимагають певного рівня перевірки для участі у виробництві блоків.

Підходи	Заходи протидії
Введення помилкових транзакцій: Для маніпулювання блокчейном або створення фальшивих транзакційних потоків.	Обмеження створення нових вузлів без довірених засвідчень.

4. Повторне Використання Адрес (Address Reuse)

Механізм: Якщо користувач використовує одну і ту ж адресу для багатьох транзакцій, це дозволяє аналізувати вхідні та вихідні потоки, щоб визначити їх зв'язки.

Підходи	Заходи протидії
Аналіз розподілу коштів: Відстеження як кошти рухаються від однієї адреси до іншої.	Використання нової адреси для кожної транзакції.
Ідентифікація ключових адрес: Виявлення адрес, що часто використовуються у великих транзакціях.	Використання технік анонімізації, таких як Stealth Addresses в Monero.

Кожен з цих методів використовує різні технічні та аналітичні підходи для виявлення інформації про учасників транзакцій у блокчейн мережах, і кожен має свої методи протидії, які можуть застосовуватись для підвищення конфіденційності.

1.3. Складність проведення атак деанонімізації

Розглянемо детально складності проведення атак деанонімізації в контексті криптовалют, зосередившись на технічних та обчислювальних ресурсах, необхідних для цього.

<u>Технічні Засоби</u>

1. Аналіз трафіку мережі:
- Необхідне обладнання: Потужні мережеві аналізатори та прослушування портів, здатні обробляти великий обсяг даних.
- Програмне забезпечення: Спеціалізоване ПЗ для аналізу мережевого трафіку, таке як Wireshark або більш спеціалізовані інструменти для декриптування та аналізу протоколів.
- Навички: Глибоке розуміння мережевих протоколів, здатність ідентифікувати аномалії та патерни в трафіку.

2. Статистичний аналіз графів транзакцій:
- Обладнання: Сервери високої потужності для обробки великих наборів даних.
- Програмне забезпечення: Інструменти для роботи з великими даними, такі як Hadoop, Spark або спеціалізоване ПЗ для аналізу графів, наприклад Neo4j.
- Навички: Знання технік машинного навчання, статистики та графових баз даних для ідентифікації шаблонів взаємодій між адресами.

3. Атаки Sybil:
- Обладнання: Велика кількість обчислювальних ресурсів для створення і підтримки численних псевдо-вузлів.
- Програмне забезпечення: Автоматизовані скрипти та програми для управління великою кількістю вузлів.
- Навички: Розуміння протоколів мережі і консенсусу, здатність маніпулювати мережевими з'єднаннями.

Обчислювальні Засоби

1. Потужність обчислень:
Великомасштабні аналітичні дії вимагають значних обчислювальних потужностей, зокрема для обробки блокчейн даних, які постійно зростають.
2. Зберігання даних:
Необхідність в зберіганні великих обсягів даних для аналізу історичних транзакцій та моніторингу мережевого трафіку.
3. Швидкість обробки:
Високий рівень відповідності системи необхідний для оперативного аналізу і реагування на зміни у блокчейн мережах.

Складнощі та Виклики

1. Масштабованість: Зі зростанням блокчейн мереж або збільшенням кількості транзакцій, необхідність в обчислювальних та технічних ресурсах зростає експоненціально.
2. Конфіденційність: Методи деанонімізації часто стикаються з юридичними та етичними питаннями, пов'язаними з приватністю користувачів.
3. Технологічний прогрес: З появою нових технологій анонімізації та криптографії методи деанонімізації повинні постійно адаптуватися і вдосконалюватися.

Ці аспекти підкреслюють складність і ресурсомісткість проведення ефективних атак деанонімізації, а також підкреслюють важливість балансу між анонімністю та прозорістю в цифрових фінансах.

1.4. Ефективність анонімних криптовалют порівняно із Bitcoin/Litecoin

Розглянемо детально втрату ефективності анонімних криптовалют у порівнянні з Bitcoin або Litecoin, фокусуючись на чотирьох ключових аспектах: швидкість транзакцій, вартість транзакцій, масштабованість мережі, та загальна прийнятність криптовалюти.

1. Швидкість транзакцій

Bitcoin і Litecoin:
- Bitcoin та Litecoin, як правило, здійснюють транзакції відносно швидко (в середньому 10 хвилин для Bitcoin і 2.5 хвилини для Litecoin на блок).
- Такі системи оптимізовані для ефективності з використанням простіших механізмів перевірки.

Анонімні криптовалюти (наприклад, Monero, Zcash):
- Анонімні криптовалюти використовують складніші алгоритми для забезпечення конфіденційності, такі як Ring Signatures в Monero або zk-SNARKs в Zcash, що може сповільнити процес обробки транзакцій.
- Ці додаткові криптографічні операції вимагають більше часу для верифікації, що може збільшувати час блоку.

2. Вартість транзакцій

Bitcoin і Litecoin:
- Транзакційні витрати можуть змінюватися, але взагалі вважаються розумними для більшості користувачів. Витрати варіюються в залежності від завантаження мережі.

Анонімні криптовалюти:
- Більш складні криптографічні процеси в анонімних криптовалютах можуть призвести до вищих транзакційних витрат. Зокрема, додаткові дані, які треба включати в кожну транзакцію, збільшують розмір транзакцій і відповідно вартість обробки.
- Це може бути особливо значущим під час періодів високого завантаження мережі.

3. Масштабованість мережі

Bitcoin і Litecoin:
- Обидві криптовалюти зіткнулися з викликами масштабованості, особливо Bitcoin зі своїм обмеженим розміром блоку (1MB).
- Рішення, такі як SegWit і Lightning Network, були розроблені для поліпшення масштабованості шляхом зменшення навантаження на головний блокчейн.

Анонімні криптовалюти:
- Анонімні криптовалюти стикаються з додатковими викликами у масштабованості через складність їхніх транзакцій. Наприклад, Monero періодично змінює свій мінімальний розмір кільця (ring size), що впливає на загальний розмір даних в блокчейні.
- zk-SNARKs, хоча і забезпечують високий рівень анонімності, мають складну структуру та високі вимоги до обчислень, що може ускладнювати швидке масштабування.

4. Загальна прийнятність

Bitcoin і Litecoin:
- Широка прийнятність в якості платіжних систем та інвестицій, що забезпечується їхньою простотою та широким визнанням.

Анонімні криптовалюти:
- Через свої анонімні властивості, такі криптовалюти часто піддаються критиці або навіть заборонам у деяких країнах через занепокоєння щодо їх використання в незаконних операціях.
- Це може обмежити їх прийнятність серед законодавчих і регуляторних органів, а також звичайних користувачів і бізнесу, які прагнуть до юридичної чистоти своїх операцій.

Ці аспекти підкреслюють те, що покращення анонімності в криптовалютах часто йде на шкоду іншим показникам, таким як швидкість, вартість і масштабованість. Такі компроміси є важливими факторами при виборі криптовалюти для використання або інвестування.

II. Для оцінки та обґрунтування необхідних ресурсів, таких як газ та ефір, необхідних для функціонування смарт-контракту на платформі Ethereum, важливо зрозуміти декілька ключових аспектів:

2.1. Розуміння Газу і Ефіру

Ефір є внутрішньою криптовалютою Ethereum, яка використовується для проведення транзакцій і взаємодії зі смарт-контрактами.

Газ — це внутрішній розрахунковий механізм для вимірювання вартості виконання операцій у Ethereum. Кожна операція або виклик смарт-контракту вимагає певної кількості газу, залежно від складності виконуваних дій.

2.2. Вартість Газу

Вартість газу вимірюється в gwei, де 1 $gwei = 10^{-9}$ ефіру. Ціна газу змінюється залежно від завантаженості мережі. Розрахунок вартості транзакції в ефірі визначається за формулою:

Вартість транзакції (в ефірах) = (Кількість газу × Ціна газу)

2.3. Оцінка Газу для Смарт-Контракту

В Ethereum, операції з смарт-контрактами можуть бути розділені на дві основні категорії:

1. Читання даних (view or pure functions):
- Ці функції не змінюють стан блокчейна і можуть бути виконані локально на вузлі без необхідності здійснення транзакції, тому вони не витрачають газ при виклику ззовні.
- Проте, якщо ці функції викликаються з інших функцій, які змінюють стан, вони впливають на загальну кількість газу, що використовується.

2. Зміна стану (state-changing operations):
- Ці функції змінюють дані в блокчейні, тому кожен такий виклик вимагає мережевої транзакції та відповідно витрат газу.
- Вартість газу визначається складністю операцій і кількістю змінних, що обробляються.

Комплексність функцій смарт-контрактів визначається на основі наступних параметрів:

- Цикли (Loops): Кожна ітерація циклу вимагає обчислень, що витрачає газ. Чим більше ітерацій, тим вища вартість.
- Умовні оператори (Conditional statements): Додавання логіки рішень, таких як `if` або `switch`, може збільшити вартість газу, особливо якщо вони впливають на виконання дорогих операцій.
- Складні обчислення: Використання математичних функцій та обрахунки, особливо з використанням чисел з плаваючою комою (що емулюється в смарт-контрактах), може суттєво збільшити вартість.

Розмір даних, що передаються в смарт-контракт, має безпосередній вплив на витрати газу:
- Зберігання даних (Storage): Вартість зберігання даних є однією з найбільших статей витрат у смарт-контрактах. Ethereum вимагає високих витрат газу за зберігання, щоб запобігти надмірному використанню блокчейн пам'яті.

- Передача даних (Data transmission): Більші обсяги вхідних даних (наприклад, масиви чи довгі рядки) збільшують вартість газу, оскільки потрібно більше ресурсів для обробки та передачі цих даних.

Для оцінки витрат газу для нового смарт-контракту можна використовувати наступні підходи:

1. Тестування та оптимізація: Розробники повинні активно тестувати смарт-контракти в тестових мережах (наприклад, Rinkeby або Ropsten), використовуючи інструменти як Remix IDE для моніторингу витрат газу.
2. Симуляція різних сценаріїв: Симулювання викликів функцій з різними вхідними даними допомагає виявити, як зміни в даних впливають на вартість газу.

Застосування цих підходів дозволить ефективно планувати і оптимізувати використання ресурсів в смарт-контрактах, забезпечуючи їхню економічну ефективність та ширшу прийнятність.

2.4.Приклад

Розглянемо приклад смарт-контракту, який реєструє платіж в блокчейн Ethereum, а також наведемо додаткові приклади різних типів функцій та їх впливу на витрати газу.

Приклад 1: Смарт-контракт для реєстрації платежів

Припустимо, смарт-контракт має функцію, яка зберігає інформацію про платежі, включно з адресою платника і сумою платежу. Ось як може виглядати код такої функції в Solidity

```
pragma solidity ^0.8.0;

contract PaymentProcessor {
    struct Payment {
        address payer;
        uint amount;
        string description;
    }

    Payment[] public payments;

    function registerPayment(address _payer, uint _amount, string memory
_description) public {
        payments.push(Payment(_payer, _amount, _description));
    }
}
```

Аналіз витрат газу:
- Запис адреси і суми: Запис адреси вимагає 20,000 газу для нових змінних (якщо це перший запис адреси в контракт), а запис числа (uint) — 20,000 газу.
- Зберігання текстового опису: Вартість зберігання рядків (строк) залежить від їх довжини. Вартість газу для зберігання є 20,000 за перше слово та 5,000 за кожне наступне слово.

Оптимізація:
- Уникайте зберігання несуттєвих текстових описів або зменшіть їх довжину.
- Розгляньте можливість агрегації платежів поза ланцюгом та записування їх як один запис для зменшення кількості транзакцій.

Приклад 2: Смарт-контракт з обрахунком комісій

Розглянемо смарт-контракт, який автоматично обраховує і вираховує комісію з транзакцій:

```
pragma solidity ^0.8.0;

contract FeeCalculator {
    uint constant public FEE_PERCENT = 1; // 1%

    function calculateFee(uint _amount) public pure returns (uint) {
        return _amount * FEE_PERCENT / 100;
    }

    function processTransaction(uint _amount) public returns (uint) {
        uint fee = calculateFee(_amount);
        uint amountAfterFee = _amount - fee;
        // Логіка для переказу коштів або їх збереження
        return amountAfterFee;
    }
}
```

Аналіз витрат газу:
- Обрахунок комісії: Математичні операції, як множення і ділення, використовують порівняно менше газу (наприклад, 5 газу за операцію).
- Функція `processTransaction`: Загальні витрати на газ залежать від додаткових операцій всередині функції, включаючи зберігання і відправку коштів.

Оптимізація:
- Використовуйте бібліотеки для безпечних математичних обчислень, такі як OpenZeppelin's SafeMath, для оптимізації витрат газу та запобігання переповнень.
- Кешуйте результати обчислень, якщо вони будуть використовуватися повторно, для зменшення загальних витрат на газ.

Ці приклади демонструють, як різні види функцій та операцій впливають на вартість газу в смарт-контрактах і як можлива оптимізація для зниження цих витрат.

2.5. Оптимізація Смарт-Контрактів

Оптимізація смарт-контрактів для зниження витрат на газ є ключовою для розробників, які прагнуть до ефективності та економічності їх виконання у мережі Ethereum. Ось детальний огляд трьох основних стратегій оптимізації:

1. Ефективне кодування

Вибір правильних типів даних:
- Використання `uint256` за замовчуванням може не завжди бути найкращим варіантом, особливо якщо значення можуть обійтися меншим розміром даних. Використання `uint8`, `uint16`, `uint32` може зменшити використання газу, коли це можливо.
- Злиття декількох булевих значень у одне `uint` може зекономити газ, оскільки кожна змінна типу `bool` займає цілий слот пам'яті (256 біт).

Мінімізація викликів до зовнішніх контрактів:
- Виклики до інших смарт-контрактів є дуже дорогими за газом. Спробуйте мінімізувати такі виклики, кешуючи необхідні дані в контракті, якщо це безпечно та можливо.

2. Уникнення непотрібних транзакцій

Перевірка умов:
- Забезпечення того, щоб транзакції не виконувались без необхідності, є критично важливим. Наприклад, перевірка умов перед тим, як змінювати стан або відправляти кошти, може запобігти непотрібним витратам на газ.

Оптимізація логіки:
- Складні функції можна розбити на дрібніші частини, які виконують логіку тільки коли це дійсно необхідно. Це може знизити кількість обчислень і, відповідно, вартість газу.

3. Сегрегація дорогих операцій

Розділення функцій:
- Розділіть великі функції на менші, які можуть бути викликані окремо, коли це потрібно. Це дозволяє користувачам платити за газ тільки тоді, коли вони дійсно використовують певні операції.

Батч-операції:
- Замість того, щоб виконувати багато однотипних транзакцій окремо, розробіть методи, які об'єднують кілька дій в одну транзакцію. Це може значно знизити загальну кількість транзакцій і, відповідно, витрати на газ.

Використання бібліотек:
- Використовуйте бібліотеки, такі як OpenZeppelin, для загальновживаних функцій, таких як безпечні математичні обчислення та стандартизовані токени. Це не тільки знижує вартість розробки, але й оптимізує витрати на газ.

Тестування та оптимізація:
- Використовуйте інструменти, такі як Remix, Truffle або Hardhat, для тестування смарт-контрактів і точного вимірювання витрат газу. Аналізуйте результати і постійно шукайте шляхи для покращення.

Ефективне використання ресурсів у смарт-контрактах не тільки знижує вартість для користувачів але й забезпечує вищу продуктивність та швидкість виконання. Тому планування та оптимізація смарт-контрактів є ключовими для успішного впровадження проектів на базі Ethereum.