

Національний технічний університет України «Київський політехнічний інститут  
імені Ігоря Сікорського»  
Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут  
імені Ігоря Сікорського»  
Міністерство освіти і науки України

Кваліфікаційна наукова  
праця на правах рукопису

КОЛІСНІЧЕНКО ВАДИМ ЮРІЙОВИЧ

УДК 004.4:004.9

**ДИСЕРТАЦІЯ**

**МЕТОДИ ТА ПРОГРАМНІ ЗАСОБИ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ**

121 Інженерія програмного забезпечення

12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів  
і текстів інших авторів мають посилання на відповідне джерело

\_\_\_\_\_ В. Ю. Колісніченко

Науковий керівник: Дорогий Ярослав Юрійович, доктор технічних наук, доцент

Київ – 2024

## АНОТАЦІЯ

*Колісниченко В. Ю.* Методи та програмні засоби аналізу блокчейн транзакцій.  
– Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії з галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення. – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, 2024.

Дисертаційна робота присвячена розробці методів та засобів для аналізу блокчейн транзакцій. Розроблені методи можуть бути прямо або опосередковано застосовані як при аналізі транзакцій, так і для дослідження блокчейн мереж загалом.

Аналіз блокчейн мереж, а саме блокчейн транзакцій, є важливою задачею в блокчейн індустрії. У цьому контексті зосередження уваги на блокчейн транзакціях є не лише академічним інтересом але й практичною необхідністю для забезпечення безпеки, ефективності та стійкості мережі.

Розробники постійно працюють над відлагодженням роботи блокчейн-вузлів, що є надзвичайно важливим для підтримки надійності мережі. Це включає в себе не тільки виявлення та усунення помилок у коді, а й оптимізацію продуктивності та масштабованості. В такому контексті аналіз транзакцій допомагає ідентифікувати та вирішувати проблеми, які можуть впливати на роботу вузлів.

Дослідники використовують аналіз транзакцій для виявлення потенційно неефективних місць у мережі. Це включає в себе вивчення шаблонів трафіку, обробку запитів, а також аналіз затримок і пропускну здатності. Такий аналіз може виявити вузькі місця, що потребують оптимізації, або навіть передбачити потенційні проблеми, які можуть виникнути в майбутньому.

Виявлення вразливостей через аналіз блокчейн транзакцій є важливим як для забезпечення безпеки мережі, так і для дослідження минулих атак. Це охоплює відстеження підозрілих або нестандартних транзакцій, що може вказувати на спроби шахрайства, викрадення коштів, атаки типу "відмова в обслуговуванні"

(DoS) або інші безпекові загрози. Аналіз транзакцій стає ключовим інструментом у боротьбі з кіберзлочинністю та забезпеченні довіри до блокчейн технологій.

**Метою дисертаційної роботи** є підвищення ефективності програмних засобів аналізу блокчейн транзакцій за рахунок удосконалення та розробки нових методів аналізу.

**У першому розділі** дисертаційної роботи проаналізовано засоби та методи дослідження блокчейн транзакцій. Розглянуто області застосування такого аналізу у різних сферах та виділено основні групи задач, які він намагається вирішити: зв'язування адреси з особою, аналіз великих потоків транзакцій та аналіз смарт-контрактів. Проведено порівняльний аналіз сучасних систем аналізу блокчейн транзакцій, серед яких є блокчейн-провідники та комплексні платформи дослідження транзакцій. Проведено огляд існуючих методів аналізу та їх обмежень, які вирішуються у даному дисертаційному дослідженні.

**У другому розділі** розроблено алгоритмічне забезпечення для аналізу блокчейн транзакцій, а саме, розроблено чотири методи, які допомагають здійснювати аналіз транзакцій: метод застосування блокчейн-провідників для отримання даних з багатьох блокчейн мереж, метод виявлення та ідентифікації блокчейн вузлів, метод виявлення джерела транзакцій та метод аналізу OP\_RETURN-скриптів.

**Третій розділ** присвячено розробці архітектури комплексної системи аналізу блокчейн транзакцій, що здатна підтримувати різні типи блокчейн мереж та дозволяти вбудовувати додаткові компоненти аналізу. У розділі розглядається процес розроблення прототипів представлених методів та їх інтеграцію до комплексної аналітичної платформи.

**Четвертий розділ** містить результати експериментального дослідження ефективності розроблених методів за допомогою реалізованих прототипів: методу застосування блокчейн-провідників для отримання даних з багатьох блокчейн мереж, методу виявлення та ідентифікації блокчейн вузлів, методу виявлення джерела транзакцій та методу аналізу OP\_RETURN-скриптів.

У дисертаційній роботі отримано низку **нових наукових результатів**, зокрема **уперше** розроблено архітектуру програмного забезпечення системи аналізу транзакцій блокчейн-мереж, яка відрізняється від існуючих застосуванням принципу інверсії керування, де компоненти аналізу окремих мереж ініціюють зв'язок та самостійно надсилають дані до ядра системи, що надає можливість інтеграції нових блокчейн-мереж та методів аналізу до системи.

**Вдосконалено** метод отримання даних з блокчейн-мереж, який на відміну від наявних способів прямого отримання даних з вузлів передбачає застосування блокчейн-провідників, що надає можливість отримувати офчейн дані, які зберігаються провідником, а також уникнути розгортання власних блокчейн-вузлів під час аналізу різних мереж.

**Вперше** здійснено формалізацію протоколу Peer Discovery блокчейн-мережі Rootstock на основі аналізу вихідного коду вузла RSKj, яка на відміну від інших включає та описує формати та послідовності повідомлень, що надає можливості для реалізації незалежних клієнтів мережі, подальшого аналізу та оптимізації даної децентралізованої системи.

**Вперше** розроблено метод обходу вузлів блокчейн-мереж, який відрізняється від вже існуючих методів можливістю роботи з блокчейн-мережею Rootstock, що дозволяє шляхом її представлення у вигляді орієнтованого графа та послідовного опитування кожного нового виявленого вузла отримати усі доступні блокчейн вузли мережі Rootstock та її структуру.

**Вперше** розроблено метод визначення відправника транзакцій у блокчейн-мережі Rootstock через підключення до кожного знайденого вузла та аналізу часу отримання нових транзакцій, який на відміну від аналогічних методів враховує та використовує особливості мережі Rootstock, що надає можливість ідентифікувати вузол (отримати ідентифікатор та IP-адресу), який першим транслиував транзакцію.

**Вдосконалено** методологію аналізу транзакцій блокчейн-мережі Bitcoin, що на відміну від наявних підходів дозволяє автоматично виділяти різні типи OP\_RETURN-скриптів на основі частоти появи їх префіксних частин та без попереднього знання форматів, що надає можливість розпізнавати та

класифікувати дані, що зберігаються або протоколи, які побудовані з використанням OP\_RETURN-скриптів.

Запропоновані методи дозволяють підвищити ефективність аналізу блокчейн транзакцій у різних сферах, таких як розробка блокчейн систем, аудит смарт контрактів, розслідування злочинів пов'язаних з блокчейн мережами, трейдинг та інше.

**Основні результати дисертаційної роботи опубліковані у 4 наукових працях, зокрема, у 4 наукових статтях, з яких 2 статті опубліковано у 2 виданнях третього квартиля (Q3), які реферуються базою Scopus, 1 статті у виданні четвертого квартиля (Q4), який реферуються базою Scopus, та 1 стаття опублікована у науковому виданні, що входить до наукових фахових видань України.**

Запропоновані методи та прототипи впроваджені та використовуються розробниками блокчейн мережі Rootstock для аналізу динаміки вузлів та стану мережі та для аналізу безпеки.

**Ключові слова:** прикладне програмне забезпечення, децентралізовані системи, розподілені бази даних, блокчейн платформи, криптовалюта, системи блокчейн аналізу, інформаційна технологія аналізу, аналітична платформа, архітектура програмного забезпечення, архітектура застосунків, проектування програмного забезпечення, аналіз транзакцій, блокчейн криміналістика, конфіденційність у блокчейн мережах, протокол пошуку вузлів.

## SUMMARY

*Kolisnichenko V.* Methods and software tools for analysis of blockchain transactions. – Qualifying scientific work, the manuscript.

Ph.D. thesis in the field of knowledge 12 Information technologies in a specialty 121 Software engineering. – National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, 2024.

The dissertation is devoted to the development of methods and tools for the analysis of blockchain transactions. The developed methods can be directly or indirectly applied both to the analysis of transactions and to the study of blockchain networks in general.

Analysis of blockchain networks, namely blockchain transactions, is an important task in the blockchain industry. In this context, focusing on blockchain transactions is not just an academic interest, but also a practical necessity to ensure the security, efficiency and stability of the network.

Developers are constantly working on debugging the operation of blockchain nodes, which is extremely important for maintaining the reliability of the network. This includes not only finding and fixing bugs in the code, but also optimizing performance and scalability. In this context, transaction analysis helps to identify and solve problems that can affect the operation of nodes.

Researchers use transaction analysis to identify potentially inefficient locations in the network. This includes studying traffic patterns, processing requests, and analyzing latency and throughput. Such analysis can reveal bottlenecks that need optimization or even predict potential problems that may arise in the future.

Vulnerability detection through analysis of blockchain transactions is important for network security and for investigating past attacks. This includes tracking suspicious or unusual transactions that may indicate attempted fraud, embezzlement, denial-of-service (DoS) attacks, or other security threats. Transaction analysis is becoming a key tool in the fight against cybercrime and ensuring trust in blockchain technology.

**The purpose** of the dissertation is to improve the effectiveness of software tools for analyzing blockchain transactions by improving and developing new algorithmic and software methods.

**The first section** of the dissertation analyzes the means and methods of blockchain transaction research. The areas of application of such analysis in various areas are considered and the main groups of problems that it tries to solve are highlighted: linking an address with a person, analysis of large transaction flows, and analysis of smart contracts. A comparative analysis of modern blockchain transaction analysis systems was conducted, including blockchain explorers and complex transaction research platforms. An overview of the existing methods of analysis and their limitations, which are addressed in this dissertation study, was conducted.

**In the second** section, algorithmic support for blockchain transaction analysis is developed, namely, four methods are developed to help perform transaction analysis: a method for using blockchain explorers to obtain data from multiple blockchain networks, a method for detecting and identifying blockchain nodes, a method for identifying the source of transactions, and a method analysis of OP\_RETURN transactions.

**The third section** is devoted to the development of the architecture of a comprehensive blockchain transaction analysis system capable of supporting various types of blockchain networks and allowing the integration of additional analysis components. The section considers the process of developing prototypes of the presented methods and their integration into a complex analytical platform.

**The fourth section** contains the results of an experimental study of the effectiveness of the developed methods with the help of implemented prototypes: the method of using blockchain explorers to obtain data from multiple blockchain networks, the method of detecting and identifying blockchain nodes, the method of detecting the source of transactions, and the method of analyzing OP\_RETURN transactions.

In the dissertation, a number of **new scientific results** were obtained, in particular, a method was developed **for the first time** the software architecture of the blockchain transaction analysis system was developed for the first time, which differs from the existing ones by applying the inversion of control principle, where the analysis components of each network initiate communication and independently send data to the system core, which provides the possibility of integration new blockchain networks and analysis methods to the system.

The method of obtaining data from blockchain networks has been **improved**, which, unlike existing methods of directly obtaining data from nodes, involves the use of blockchain explorers, which provides the opportunity to receive off-chain data stored by the explorers, as well as to avoid the deployment of own blockchain nodes during analysis various networks.

**For the first time**, the formalization of the Peer Discovery protocol of the Rootstock blockchain network was carried out based on the analysis of the source code of the RSKj node, which, unlike others, includes and describes message formats and sequences, which provides opportunities for the implementation of independent network clients, further analysis and optimization of this decentralized system.

**For the first time**, a method of nodes traversing of blockchain networks has been developed, which differs from existing methods by the possibility of working with the Rootstock blockchain network, which allows, by presenting it in the form of a directed graph and sequentially polling each newly discovered node, to obtain all available blockchain nodes of the Rootstock network and its structure.

**For the first time**, a method of identifying the sender of transactions in the Rootstock blockchain network by connecting to each found node and analyzing the time of receiving new transactions has been developed, which, unlike similar methods, takes into account and uses the features of the Rootstock network, which provides the opportunity to identify a node (get an ID and an IP address), which was the first to broadcast the transaction.

The methodology for analyzing transactions of the Bitcoin blockchain network has been **improved**, which, unlike existing approaches, allows to automatically distinguish different types of OP\_RETURN scripts based on the frequency of appearance of their prefix parts and without prior knowledge of the formats, which provides the opportunity to recognize and classify the data stored or the protocols that built using OP\_RETURN scripts. The proposed methods and prototypes make it possible to increase the effectiveness of the analysis of blockchain transactions in various areas, such as the development of blockchain systems, audit of smart contracts, investigation of crimes related to blockchain networks, trading, and others.



The main results of the dissertation work **were published** in 4 scientific papers, in particular, in 4 scientific articles, of which 2 articles were published in 2 editions of the third quartile (Q3) referenced by the Scopus database, 1 article in the edition of the fourth quartile (Q4) referenced by the Scopus database, and 1 article was published in a scientific publication, which is included in the scientific specialized publications of Ukraine.

The proposed methods and prototypes are implemented and used by the developers of the Rootstock blockchain network to analyze the dynamics of nodes and the state of the network, and to analyze security.

**Keywords:** application software, decentralized systems, distributed databases, blockchain platforms, cryptocurrency, blockchain analysis systems, information technology of analysis, analytical platform, software architecture, applications architecture, software design, transaction analysis, blockchain forensics, privacy in blockchain networks, peer discovery protocol.

## Список публікацій здобувача

*Наукові праці, в яких опубліковані основні наукові результати дисертації:*

1. Dorogyu, Y., Kolisnichenko, V. Blockchain Transaction Analysis: A Comprehensive Review of Applications, Tasks and Methods. *System research and information technologies*. 2023. Vol. 4. P. 37–53. (Scopus Q4).
2. Dorogyu, Y., Kolisnichenko, V. Devising a method for rapid data retrieval using explorers for blockchain analysis. *Eastern-European Journal of Enterprise Technologies*. 2023. Vol. 4, No. 2 (124). P. 6–16. (Scopus Q3, фахове видання категорії «А»).
3. Дорогий Я. Ю., Колісніченко В.Ю. Застосування логування різними учасниками блокчейн-мереж для деанонізації кінцевого користувача. *Вісник Хмельницького національного університету. Серія: «Технічні науки»*. 2023. Т. 1, №5. С. 60–66. (Фахове видання категорії «Б»)
4. Dorogyu, Y., Kolisnichenko, V. Developing a method for the detection and identification of rootstock blockchain network nodes. *Eastern-European Journal of Enterprise Technologies*. 2024. Vol. 1, No. 2 (127). P. 6–15. (Scopus Q3, фахове видання категорії «А»).

## ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ .....	14
ВСТУП.....	15
РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ.....	19
1.1 Аналіз предметної галузі блокчейн-аналізу.....	19
1.2 Аналіз існуючих методів та інструментів для аналізу блокчейн транзакцій.....	29
1.2.1 Асоціювання адреси з особою.....	29
1.2.2 Аналіз потоків транзакцій.....	30
1.2.3 Аналіз смарт-контрактів .....	33
1.3 Порівняльний аналіз сучасних систем аналізу блокчейн транзакцій.....	35
1.3.1 Блокчейн провідники .....	35
1.3.2 Комплексні системи дослідження блокчейн мереж.....	36
1.3.3 Інші засоби дослідження.....	38
1.4 Огляд обмежень існуючих методів та постановка завдань дисертаційного дослідження.....	38
1.4.1 Збір даних з блокчейн-мереж для аналізу транзакцій.....	38
1.4.2 Деанонімізація користувача на мережевому рівні .....	41
1.4.3 Виявлення вузлів блокчейн-мереж .....	46
1.4.4 Ідентифікація відправника транзакцій .....	49
1.4.5 Bitcoin скрипти та OP_RETURN опкод.....	50
1.4.4 Постановка завдань .....	51
1.5 Висновки до розділу 1 .....	53
РОЗДІЛ 2 АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ МЕТОДІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ .....	54
2.1 Метод застосування блокчейн-провідників для збору даних з різних мереж .....	54
2.1.1 Основні етапи методу.....	54
2.1.2 Вибір блокчейн-провідника.....	55

2.1.3	Схема збору даних для аналізу блокчейн мереж.....	57
2.1.4	Формалізоване представлення методу .....	60
2.2	Метод виявлення та ідентифікації блокчейн вузлів .....	61
2.2.1	Особливості Peer Discovery протоколу.....	61
2.2.2	Основні етапи методу.....	65
2.2.3	Формалізоване представлення методу .....	66
2.3	Метод визначення відправника транзакцій за допомогою псевдовузлів .....	69
2.3.1	Особливості Wire протоколу .....	69
2.3.2	Основні етапи методу.....	72
2.3.3	Формалізоване представлення методу .....	74
2.4	Метод аналізу OP_RETURN-скриптів Bitcoin мережі .....	75
2.4.1	Особливості OP_RETURN-скриптів Bitcoin мережі .....	75
2.4.2	Основні етапи методу.....	76
2.4.5	Формалізоване представлення методу .....	77
2.5	Висновки до розділу 2 .....	79
РОЗДІЛ 3 ЗАСОБИ РОЗРОБКИ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ.....		81
3.1	Архітектура програмної системи аналізу блокчейн транзакцій .....	81
3.2	Засоби розробки .....	83
3.3	Програмне забезпечення застосування блокчейн-провідників для збору даних з різних мереж .....	84
3.4	Програмне забезпечення виявлення та ідентифікації блокчейн вузлів.....	87
3.5	Програмне забезпечення визначення відправника транзакцій за допомогою псевдовузлів .....	88
3.6	Програмне забезпечення аналізу OP_RETURN-скриптів Bitcoin мережі....	89
3.7	Висновки до розділу 3 .....	91
РОЗДІЛ 4 ЕКСПЕРИМЕНТАЛЬНИЙ АНАЛІЗ РОЗРОБЛЕНИХ МЕТОДІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ .....		93
4.1	Оцінка ефективності застосування блокчейн-провідників для збору даних з різних мереж.....	93

4.1.1 Параметри експерименту .....	93
4.1.2 Результати .....	93
4.1.3 Обговорення результатів .....	101
4.2 Оцінка ефективності виявлення та ідентифікації блокчейн вузлів .....	104
4.2.1 Параметри експерименту .....	104
4.2.2 Результати .....	105
4.2.3 Обговорення результатів .....	108
4.3 Оцінка ефективності визначення відправника транзакцій за допомогою псевдовузлів .....	112
4.3.1 Параметри експерименту .....	112
4.3.2 Результати .....	113
4.3.3 Обговорення результатів .....	114
4.4 Оцінка ефективності аналізу OP_RETURN-скриптів Bitcoin мережі .....	115
4.4.1 Параметри експерименту .....	115
4.4.2 Результати .....	116
4.4.3 Обговорення результатів .....	119
4.5 Висновки до розділу 4 .....	119
ВИСНОВКИ .....	121
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	124
ДОДАТОК А .....	143
ДОДАТОК Б.....	144

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AML – Anti-Money Laundering (боротьба з відмиванням грошей)

API – application programming interface (прикладний програмний інтерфейс)

BTC – Bitcoin

DeFi – Decentralized finance (децентралізовані фінансові сервіси)

ETH – Ethereum

EVM – Ethereum Virtual Machine

JSON-RPC – JavaScript Object Notation (JSON) Remote Procedure Call (RPC)

KYC – Know Your Customer (знай свого клієнта)

NFT – Non-fungible token (взаємозамінний токен)

OP\_RETURN – операційний код, що позначає транзакцію як недійсну та дозволяє включення додаткових даних до транзакції

RSK – Rootstock

UTXO – Unspent Transaction Output (невитрачений вихід транзакції)

Ончейн – активність, яка відбувається у блокчейні

Офчейн – активність, яка відбувається поза блокчейном

Сайдчейн – блокчейн мережа, пов'язана з певною основною мережею

## ВСТУП

**Актуальність теми.** Аналіз блокчейн мереж, а саме, блокчейн транзакцій є важливою задачею в блокчейн індустрії. У цьому контексті зосередження уваги на блокчейн транзакціях є не лише академічним інтересом, а й практичною необхідністю для забезпечення безпеки, ефективності та стійкості мережі.

Розробники постійно працюють над відлагодженням роботи блокчейн-вузлів, що є надзвичайно важливим для підтримки надійності мережі. Це включає в себе не тільки виявлення та усунення помилок у коді, а й оптимізацію продуктивності та масштабованості. В такому контексті аналіз транзакцій допомагає ідентифікувати та вирішувати проблеми, які можуть впливати на роботу вузлів.

Дослідники використовують аналіз транзакцій для виявлення потенційно неефективних місць у мережі. Це включає в себе вивчення шаблонів трафіку, обробку запитів, а також аналіз затримок і пропускну здатності. Такий аналіз може виявити вузькі місця, що потребують оптимізації, або навіть передбачити потенційні проблеми, які можуть виникнути в майбутньому.

Виявлення вразливостей через аналіз блокчейн транзакцій є важливим як для забезпечення безпеки мережі, так і для дослідження минулих атак. Це охоплює відстеження підозрілих або нестандартних транзакцій, що може вказувати на спроби шахрайства, викрадення коштів, атаки типу "відмова в обслуговуванні" (DoS) або інші безпекові загрози. Аналіз транзакцій стає ключовим інструментом у боротьбі з кіберзлочинністю та забезпеченні довіри до блокчейн технологій.

### **Зв'язок роботи з науковими програмами, планами, темами.**

Тема дисертаційної роботи відповідає планам науково-дослідної та навчальної роботи кафедри інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут». Дисертаційна робота розпочата на кафедрі інформаційних систем та технологій, завершена на кафедрі інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» і виконувалась в рамках наступної НДР: «Хмарна платформа

розроблення і управління функціонуванням критичних IT-інфраструктур, що опрацьовують великі обсяги даних» (номер державної реєстрації № 0220U100804).

**Мета і завдання дослідження.** Метою дисертації є підвищення ефективності програмних засобів аналізу блокчейн транзакцій за рахунок удосконалення та розробки нових методів аналізу.

Для досягнення мети сформовані такі **завдання**:

- проаналізувати стан предметної галузі аналізу блокчейн мереж;
- розробити метод та програмне забезпечення для збору даних з різних блокчейн мереж;
- розробити метод та програмне забезпечення виявлення та ідентифікації вузлів блокчейн мережі Rootstock;
- розробити метод та програмне забезпечення ідентифікації відправника транзакцій блокчейн мережі Rootstock;
- розробити метод та програмне забезпечення аналізу OP\_RETURN-скриптів блокчейн мережі Bitcoin;
- запропонувати архітектуру системи аналізу блокчейн транзакцій та інтеграцію розроблених методів до такої системи.

**Об’єкт дослідження** – блокчейн-мережі та транзакції.

**Предмет дослідження** – методи, алгоритми та засоби аналізу блокчейн транзакцій.

**Методи дослідження:** методи системного та порівняльного аналізу, теорія алгоритмів, теорія програмування, теорія програмних систем, теорія блокчейн-систем, методи аналізу транзакцій.

**Наукова новизна** одержаних результатів полягає у наступному:

- Вперше розроблено архітектуру програмного забезпечення системи аналізу транзакцій блокчейн-мереж, яка відрізняється від існуючих застосуванням принципу інверсії керування, де компоненти аналізу окремих мереж ініціюють зв’язок та самостійно надсилають дані до ядра системи, що надає можливість інтеграції нових блокчейн-мереж та методів аналізу до системи.



– Вдосконалено метод отримання даних з блокчейн-мереж, який на відміну від наявних способів прямого отримання даних з вузлів, передбачає застосування блокчейн-провідників, що надає можливість отримувати офчейн дані, які зберігаються провідником, а також уникнути розгортання власних блокчейн-вузлів під час аналізу різних мереж.

– Вперше здійснено формалізацію протоколу Peer Discovery блокчейн-мережі Rootstock на основі аналізу вихідного коду вузла RSKj, яка на відміну від інших включає та описує формати та послідовності повідомлень, що надає можливості для реалізації незалежних клієнтів мережі, подальшого аналізу та оптимізації даної децентралізованої системи.

– Вперше розроблено метод обходу вузлів блокчейн-мереж, який відрізняється від вже існуючих методів можливістю роботи з блокчейн-мережею Rootstock, що дозволяє шляхом її представлення у вигляді орієнтованого графа та послідовного опитування кожного нового виявленого вузла отримати усі доступні блокчейн вузли мережі Rootstock та її структуру.

– Вперше розроблено метод визначення відправника транзакцій у блокчейн-мережі Rootstock через підключення до кожного знайденого вузла та аналізу часу отримання нових транзакцій, який на відміну від аналогічних методів враховує та використовує особливості мережі Rootstock, що надає можливість ідентифікувати вузол (отримати ідентифікатор та IP-адресу), який першим транслиував транзакцію.

– Вдосконалено методологію аналізу транзакцій блокчейн-мережі Bitcoin, що на відміну від наявних підходів дозволяє автоматично виділяти різні типи OP\_RETURN-скриптів на основі частоти появи їх префіксних частин та без попереднього знання форматів, що надає можливість розпізнавати та класифікувати дані, що зберігаються або протоколи, які побудовані з використанням OP\_RETURN-скриптів.

### **Практичне значення одержаних результатів**

Запропоновані методи та прототипи дозволяють підвищити ефективність аналізу блокчейн транзакцій у різних сферах, таких як розробка блокчейн систем,

аудит смарт контрактів, розслідування злочинів пов'язаних з блокчейн мережами, трейдинг та інше.

### **Особистий внесок здобувача.**

Усі результати наукового дослідження, представлені до захисту, отримані автором особисто. У написаних у співавторстві публікаціях автору належать такі результати: [1] – огляд існуючих систем, методів та засобів аналізу блокчейн мереж, їх структуризація ; [2] – аналіз блокчейн провідників, розробка та реалізація алгоритму, проведення експерименту, [3] – аналіз вихідного коду блокчейн вузлів, структуризація їх типів, аналіз методів деанонімізації, [4] – аналіз протоколу Peer Discovery за допомогою рев'ю вихідного коду програмного забезпечення, розробка алгоритму виявлення вузлів, реалізація прототипу та проведення експерименту.

### **Апробація результатів дисертації.**

Запропоновані методи та прототипи впроваджені та використовуються розробниками блокчейн мережі Rootstock для аналізу динаміки вузлів та стану мережі та для аналізу безпеки.

**Публікації.** Основні результати дисертаційної роботи опубліковані у 4 наукових працях, зокрема, у 4 наукових статтях, з яких 2 статті опубліковано у 2 виданнях третього квартиля (Q3), які реферуються базою Scopus, 1 статті у виданні четвертого квартиля (Q4), який реферуються базою Scopus, та 1 стаття опублікована у науковому виданні, що входить до наукових фахових видань України.

**Структура і обсяг роботи.** Дисертаційна робота складається зі вступу, 4 розділів, висновків, 2 додатка та списку використаних джерел із 181 найменувань. Загальний обсяг роботи становить 144 сторінки, з них 109 сторінки основного тексту, 40 рисунків, 8 таблиць.

## РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ ТА ПРОГРАМНИХ ЗАСОБІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ

### 1.1 Аналіз предметної галузі блокчейн-аналізу

Блокчейн аналіз служить ключовим інструментом для розуміння тонкощів блокчейн мереж. Область аналізу охоплює різноманітні методології та методи, спрямовані на отримання важливої інформації та деталей з даних децентралізованих систем. Такий аналіз передбачає різні аспекти: аналіз транзакцій, аналіз мережі, технічний аналіз, бізнес аналіз і т. д.. Зазвичай під аналізом блокчейн мереж розуміють дослідження транзакцій, тобто інформації, яка зберігається на блокчейні (ончейн інформація). Однак вивчення транзакцій є лише елементом більш комплексного аналізу (рис. 1.1).



Рис. 1.1. Аналіз блокчейн мереж та транзакцій

Аналіз блокчейн транзакцій відноситься до систематичного дослідження та інтерпретації даних блокчейну з метою отримання поглибленого розуміння динаміки транзакцій, розпізнавання патернів та виявлення відхилень від норми. Це інструмент, який покращує розуміння поведінки користувачів та важливих подій в блокчейн мережах. Оскільки сфери блокчейну та децентралізованих фінансів

(decentralized finance – DeFi) стрімко розвиваються, зростає і потенціал для використання аналізу транзакцій.

Блокчейн транзакцію можна визначити як запис про передачу цифрових активів в мережі блокчейн, яка постійно записується в розподілену книгу фінансового обліку. Механізми асиметричної криптографії використовуються для визначення відправника або одержувача транзакції – адреси формуються з публічних ключів, а приватні ключі використовуються для підпису транзакцій (щоб довести, що фактичний власник коштів створив транзакцію).

З точки зору ведення записів існує дві загальні моделі: модель невитрачених транзакцій (UTXO – Unspent Transaction Output) і модель облікового запису. У моделі UTXO (рис. 1.2) кожна транзакція створює список виходів, які будуть витрачені в майбутніх транзакціях (використовуються як вхідні дані). Виходи призначаються за адресами, які повинні мати можливість їх використовувати (витрачати). Загальний баланс адреси – це сума всіх невитрачених виходів на цю адресу в поточний момент.

З іншого боку, модель облікового запису набагато простіша для розуміння. Блокчейн підтримує баланс для кожного облікового запису та веде облік усіх транзакцій, які вплинули на цей баланс.

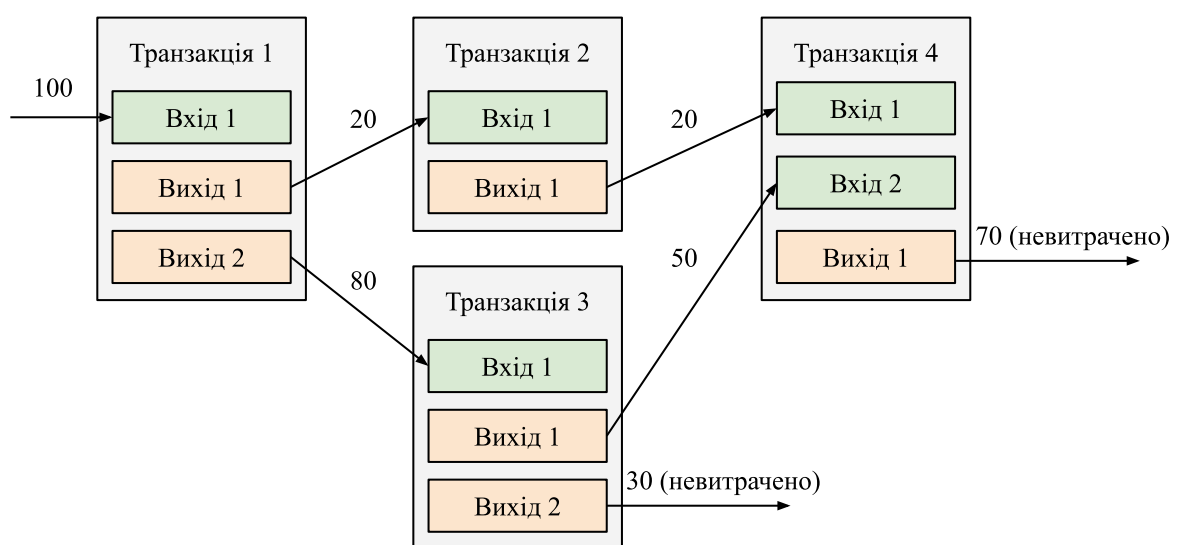


Рис. 1.2. Спрощена модель UTXO

Bitcoin-транзакції не мають полів відправника чи отримувача. Замість цього застосовуються скрипти блокування (`scriptPubKey`) і розблокування (`scriptSig`), щоб створити концепцію головоломки, яку можна вирішити за допомогою певних умов. Наприклад, щоб провести транзакцію, потрібно вказати підпис у скрипті розблокування, публічний ключ якого вказано у скрипті блокування.

Біткоїн-скрипти обмежені і не дозволяють створювати складну логіку. Для усунення даного недоліку мережа Ethereum використовує концепцію смарт-контракту [5], яка дозволяє створювати складні програми за допомогою JavaScript-подібної мови під назвою Solidity, які зберігаються та виконуються на блокчейні.

Смарт-контракти надають можливість створення програм різного рівня складності, надаючи кінцевим користувачам безпечні децентралізовані фінансові (DeFi) послуги, а розробникам – інструменти для розробки і подальшої оптимізації.

Крім ончейн, важливими також є й офчейн дані – технічні дані, які не є частиною блокчейну, і зазвичай не впливають на його бізнес-логіку. Розглянемо спрощений процес включення транзакцій в блокчейн (схожий для різних мереж). Користувач створює транзакцію та підписує її, надаючи докази власності на адресу та цілісності транзакції. Після підписання транзакції користувач транслює її за допомогою власного вузла або через інтерфейс JSON-RPC вибраного публічного вузла. Проходячи через послідовності вузлів, транзакція нарешті досягає майнерів, які включають її в блок і видобувають. Після того, як блок видобуто, він отримує трансляцію на решту вузлів. Жодна мережева інформація (IP-адреса відправника та вузлів, які транслювали транзакцію тощо) у цьому процесі не включається в блокчейн, однак проміжні хости можуть зберігати її у своїх власних базах даних.

Аналіз транзакцій блокчейну — це потужний інструмент, який дозволяє краще зрозуміти поведінку користувачів і події в мережах блокчейну. У цьому розділі розглядаються ключові аспекти застосування процесу аналізу транзакцій, включаючи дослідження криптовалют, управління ризиками, дотримання податкового законодавства та багато інших.

**Розслідування злочинів.** Криптовалюти мають такі унікальні властивості, як децентралізація, незалежність від банків, безпека, повсюдність і анонімність. Як

і з іншими видами активів, ці унікальні характеристики визначають специфічні застосування криптовалют. Проте, ці самі характеристики також зробили криптовалюту привабливими для незаконних дій, зокрема для відмивання грошей, шахрайства та ухилення від санкцій [6], [7].

Одним з найбільших прикладів шахрайства із залученням криптовалют є інцидент Africrypt. Двоє засновників Africrypt стверджували, що їхню компанію зламали, що призвело до крадіжки всіх її активів. Після заяви засновники зникли. Загалом зникло приблизно 3,6 мільярда доларів у біткоїнах [8]. На даний момент не знайдено багато додаткової інформації щодо цієї справи. Повідомляється, що правоохоронні органи продовжують пошуки засновників [9]. Цей інцидент, серед багатьох інших [10], [11], схожий на традиційне фінансове шахрайство, коли засновники (імена яких часто відомі) збирають гроші та зникають.

Хоча технології блокчейн забезпечують певний рівень конфіденційності, їх не можна вважати повністю анонімними [12], і в багатьох випадках ретельний аналіз може дати відповідь на питання «куди йдуть гроші» [13]. Розглянемо деякі відомі випадки, коли аналіз транзакцій був корисним для розслідування.

Криптовалюти часто використовуються як платіжні засоби під час кібервимагань і атак програм-вимагачів. Хакери, які здійснюють ці атаки, вимагають оплату в криптовалюті в обмін на повернення контролю над комп'ютерною системою жертви або викраденими даними. Одним із таких випадків є зловмисне програмне забезпечення NetWalker, створене за моделлю Ransomware as a Service (RaaS) [14], де філії орендують зловмисне програмне забезпечення від операторів для здійснення атак. Одну з афілійованих осіб заарештували, і за допомогою інструментів аналізу блокчейну вдалося відстежити адреси, пов'язані з цим підприємством [15].

Злам проєктів DeFi є досить поширеним [16], [17]. На відміну від інших доменів, у домені блокчейн хакер безпосередньо керує цінними активами, такими як монети чи токени. Варто зазначити, що найбільші обсяги активів зосереджені в кросчейн-мостах [18] і централізованих криптобіржах (CEX), що робить їх

привабливими цілями [19]. Аналіз транзакцій зазвичай застосовується, щоб отримати розуміння атаки та відстежити тих, хто був залучений.

Аналіз атаки є важливим заходом, який необхідно вжити після того, як стався інцидент. Цей процес передбачає визначення того, як систему було зламано, оцінку масштабу завданої шкоди, визначення стратегій для мінімізації шкоди та усунення будь-яких вразливостей, які були використані. Транзакції зловмисників разом із залученими смарт-контрактами аналізуються, щоб визначити, як систему блокчейн було зламано. Такий аналіз часто виконується компанією-власником, компанією-розслідувачем або блокчейн-спільнотою [20], [21] з різними рівнями деталізації.

Хакери, які викрадають кошти з блокчейнів, часто прагнуть відмити вкрадені криптовалюти, щоб приховати свою особу та ускладнити правоохоронним органам відстеження незаконних коштів [22]. Для цього вони використовують міксери або інші методи обфускації, щоб приховати слід транзакцій і ускладнити відстеження вкрадених коштів. Крім того, хакери можуть використовувати децентралізовані біржі для конвертації вкрадених криптовалют в інші активи, такі як конфіденційні монети або стейблкоїни, щоб ще більше заплутати сліди транзакцій. Потім ці активи можуть бути переміщені через кілька гаманців, щоб ще більше віддалити кошти від початкової крадіжки. Останній крок може включати конвертацію вкрадених криптовалют у фіатну валюту через регульований обмін або інші способи переведення в готівку незаконних коштів.

Одним з успішних розслідувань відмивання коштів є справа Bitfinex. За даними Elliptic [23], після зламу вкрадені кошти повільно відмивалися різними техніками. AlphaBay — один із сервісів, який використовувався як міксер для приховування слідів. Однак пізніше його вилучили правоохоронні органи, і це, ймовірно, дозволило їм вийти на сліди хакерів.

Ще один, менш успішний випадок розслідування — це злом біржі Zaif у вересні 2018 року. Команда інженерів Crystal Blockchain Analytics провела аналіз руху біткоїнів [24] і змогла знайти адреси, причетні до зламу. Хоча власники адрес невідомі, адреси відстежуються на випадок подальших транзакцій.

Одним із відомих типів блокчейн-активів, який привернув велику увагу, є незамінний токен (NFT). NFT — це активи, які представляють право власності на унікальні елементи, такі як музика, відео, мистецтво тощо в мережі блокчейн. Вони нероздільними та невзаємозамінними. Кожен NFT унікальний і не може бути відтворений. Хоча він може бути залучений до кримінальної діяльності, як і інші цифрові активи, важливо також відзначити проблему порушення авторських прав. Технологія блокчейн може гарантувати унікальність токена, але не може гарантувати унікальність представленого активу, який можна скопіювати. Аналіз транзакцій використовується для допомоги в захисті авторських прав на NFT, а також для перевірки його автентичності і ідентифікації оригінального творця або власника твору. Ця інформація також може бути використана як доказ для підтвердження претензій щодо авторських прав. Одним із відомих проектів, який намагається виявити порушення авторських прав шляхом сканування блокчейнів і торгових майданчиків, є DeviantArt [25]. Вони використовують різні методи, включаючи машинне навчання, щоб визначити копію.

Аналіз транзакцій блокчейну використовується як цінний інструмент у розслідуванні злочинів, що дозволяє правоохоронним органам відстежувати потік коштів у мережі блокчейн та виявляти будь-яку підозрілу діяльність, пов'язану з незаконною діяльністю, такою як відмивання грошей, транзакції в даркнеті, кіберзлочинність, і шахрайство. Аналіз транзакцій також допомагає відстежувати потік коштів, пов'язаний з кібератаками, платежами за програми-вимагачі та іншими шкідливими діями. Він надає інформацію про поведінку транзакцій і моделі, які можна використовувати для виявлення потенційної злочинної діяльності та вжиття відповідних заходів.

**Регулювання.** Положення про криптовалюту — це закони чи правила, встановлені урядами чи регулюючими органами для регулювання використання, торгівлі та зберігання криптовалют. Ці правила спрямовані на захист інвесторів, запобігання незаконній діяльності, такій як відмивання грошей і фінансування тероризму, а також сприяють стабільності та цілісності фінансової системи. Регулювання криптовалюти може охоплювати широкий спектр тем, залежно від



юрисдикції та конкретних проблем регуляторів [26]. Хоча ці правила здебільшого стосуються криптовалют, а не самих технологій, деякі країни можуть спробувати забезпечити дотримання правил і з технічної сторони (наприклад, щодо майнінгу) [27].

Аналіз транзакцій є корисним інструментом для забезпечення дотримання нормативних актів та вимог щодо криптовалют [28], [29]. Регулятори можуть використовувати аналіз транзакцій для моніторингу та виявлення потенційної діяльності з відмивання грошей, забезпечення КҮС (знай свого клієнта) та дотримання податкового законодавства, запобігання шахрайству та захисту споживачів на ринку криптовалют [30]. Таким чином, виявляючи будь-яку незвичну або підозрілу діяльність, регулятори мають змогу вжити відповідних заходів для запобігання відмиванню грошей та іншим фінансовим злочинам.

**Торгівля та інвестиції.** У традиційних фінансах фінансові операції здебільшого непрозорі, і інвестори часто покладаються на посередників [31], щоб надати інформацію про активи, у які вони інвестують. Інвестори та трейдери використовують такі методи, як технічний аналіз, щоб аналізувати фінансові ринки та цінні папери на основі статистичних тенденцій, та моделі історичних даних про ціни та обсяги. Торгівля в блокчейні пропонує нові можливості та виклики завдяки своїм унікальним характеристикам прозорості, безпеки та децентралізації [32].

Доступ до даних транзакцій може надати велику перевагу, за умови належної обробки величезних обсягів даних. Тому важливо створювати нові методи та інструменти, які можуть пояснити поведінку учасників ринку. Крім того, ці методи та інструменти мають бути такими ж або кращими, ніж у ваших потенційних опонентів, оскільки вони також мають доступ до тих самих вихідних даних.

Аналіз блокчейн-транзакцій надає інформацію про розподіл цифрових активів між учасниками ринку та про їхню поведінку. Ця інформація може допомогти трейдерам та інвесторам визначити потенційні рівні цін на певну криптовалюту на основі рівня попиту з боку покупців і продавців. Таким чином трейдери та інвестори мають змогу отримати уявлення про торгові стратегії

учасників і використовувати цю інформацію для коригування власних торгових рішень.

Крім того, аналіз застосовується для дослідження потоку активів у блокчейні, щоб ідентифікувати великі транзакції та рух коштів, які можуть свідчити про маніпулювання ринком або іншу незаконну діяльність. Аналіз потоку транзакцій допомагає трейдерам уникнути входу в позиції, які можуть бути вразливими до різких коливань цін.

**Управління ризиками.** Організації, які намагаються застосувати технології блокчейн і DeFi для свого бізнесу, повинні знати про численні додаткові ризики [33], [34], [35].

Криптовалюти все ще розвиваються, і одним із основних ризиків є нечіткість нормативних актів, зокрема дотримання законодавства та нормативних вимог. Підприємства, які використовують блокчейн-технології, можуть зіткнутися з проблемами у дотриманні чинних нормативних актів або прогнозуванні майбутніх нормативних актів, що може призвести до юридичних і фінансових санкцій або шкоди репутації. Ризик нечітких нормативних актів щодо управління ризиками блокчейну є значним, оскільки технологія блокчейну працює в регуляторній сірій зоні в багатьох країнах.

Інший набір ризиків виникає з технічної сторони. Помилки, уразливості, проблеми з масштабованістю мережі можуть призвести до різних негативних наслідків, таких як втрата або крадіжка коштів [36], простій мережі [37], репутаційна шкода тощо.

Волатильність і ліквідність є ще двома значними ризиками, пов'язаними з блокчейном і криптовалютами [38]. Ці ризики можуть вплинути як на інвесторів, так і на підприємства, які використовують криптовалюти для транзакцій або інших цілей. Волатильність може призвести до значних втрат для інвесторів, які інвестували в криптовалюти, оскільки вартість їхніх інвестицій може швидко зменшуватися. Крім того, компанії, які використовують криптовалюти для транзакцій, можуть зазнати негативного впливу волатильності, оскільки вартість їхніх транзакцій також може швидко коливатися. Криптовалютні ринки можуть

бути відносно неліквідними, особливо для менш популярних криптовалют або в періоди нестабільності ринку. Ця неліквідність може ускладнити інвесторам продаж своїх криптовалют, коли їм це необхідно, що призведе до збитків. Крім того, неліквідність може створити проблеми для компаній, які використовують криптовалюту для транзакцій, оскільки може бути важко знайти покупця чи продавця бажаної криптовалюти за справедливою ринковою ціною.

Аналіз транзакцій є корисним інструментом для управління ризиками в блокчейні. Він надає компаніям уявлення про поведінку та моделі транзакцій, які можна використовувати для визначення потенційних ризиків і вразливостей. Аналіз транзакцій використовується для виявлення шахрайства, наприклад відмивання грошей або інших фінансових злочинів, шляхом аналізу моделей транзакцій і виявлення будь-яких незвичайних або підозрілих дій. Він також може допомогти компаніям контролювати дотримання правил і галузевих стандартів, виявляючи будь-які потенційні проблеми з відповідністю. Компанії можуть визначити рівень ризику, пов'язаного з конкретною операцією чи клієнтом, і вжити відповідних заходів для управління цим ризиком. Крім того, аналіз транзакцій допомагає інвесторам і підприємствам оцінити ліквідність криптовалют шляхом аналізу обсягів транзакцій.

**Управління ланцюгом поставок.** Управління ланцюгом поставок у блокчейні означає використання технології блокчейну для відстеження та керування рухом товарів і послуг через ланцюг поставок. Технології Blockchain забезпечують прозору та безпечну платформу для відстеження та перевірки транзакцій у режимі реального часу [39]. Кожен актив представлений унікальним маркером. Коли сторона виконує передачу активу, вона також створює та підписує транзакцію для передачі токена (який представляє фактичний актив) у цьому блокчейні. Потім транзакції реєструються в блокчейні, і весь процес є прозорим для акціонерів. Це може допомогти підприємствам оптимізувати свої операції в ланцюзі поставок, зменшити витрати та забезпечити відповідність відповідним нормам і галузевим стандартам.

У процесі ланцюга поставок аналіз транзакцій блокчейну є основним інструментом, який дозволяє зацікавленим сторонам стежити за всім процесом. Це дозволяє витягувати та аналізувати схеми транзакцій, що робить можливим отримати цінну інформацію про рух товарів і послуг через ланцюг поставок [40], [41], [42].

**Блокчейн розробка.** Аналіз транзакцій також важливий для розробки блокчейну та його оптимізації. На різних етапах розробки транзакції аналізуються для налагодження помилок [43] і моніторингу працездатності мережі. Цей аналіз використовується для того, щоб отримати розуміння поведінки користувачів у мережі, визначити їхні потреби та проблеми [44]. Аналізуючи моделі транзакцій, розробники можуть визначити вузькі місця в мережі блокчейну, такі як перевантажені вузли [45] або високі комісії за транзакції. Отримана інформація використовується для розробки нових рішень та оптимізації блокчейн-платформи. Така оптимізація може стосуватися його продуктивності [46], [47] або безпеки [48]. Крім того, аналіз транзакцій дозволяє розробникам виявляти підозрілу активність або DoS-атаки та вживати заходів для зменшення ризиків [49].

**Виявлення та попередження атак на блокчейн.** Аналіз транзакцій у реальному часі використовується для моніторингу смарт-контрактів, щоб виявити можливі атаки та запобігти їм. Аналіз даних транзакцій у реальному часі дає змогу виявляти будь-яку підозрілу активність, дозволяючи своєчасно втручатися, щоб запобігти або мінімізувати вплив атаки.

Одним із таких широко використовуваних рішень є Forta [50]. Він отримує переваги аналізу транзакцій для виявлення та пом'якшення загроз безпеці в децентралізованих програмах і смарт-контрактах. Технологія Forta розроблена для аналізу блокчейн-транзакцій і даних для виявлення та запобігання зламам, експлуатації та іншим зловмисним діям. Зазначається [51], що використання системи могло б запобігти численним атакам і фінансовим збиткам.

## 1.2 Аналіз існуючих методів та інструментів для аналізу блокчейн транзакцій

Основні задачі аналізу блокчейн мереж можна розділити на три великі групи (рис. 1.3): асоціювання адреси з особою, аналіз потоків транзакцій та аналіз смарт-контрактів. У цьому розділі розглянуто дані завдання та методи, які використовуються для їх вирішення.



Рис. 1.3. Основні завдання аналізу блокчейн транзакцій

### 1.2.1 Асоціювання адреси з особою

Поширеним завданням аналізу транзакцій є ідентифікація власника адреси. Під володінням адресою розуміється, що особа володіє закритим ключем (або початковою/мнемонічною фразою) і відповідним відкритим ключем, з якого створюється адреса. Адреси створюються за допомогою криптографічних механізмів до взаємодії з блокчейном. Подібно до цієї задачі є і протилежна — знайти адреси, що належать певній особі

Один із найпростіших методів зв'язати адреси з особами — вимога до користувачів розкривати свої особи під час купівлі або продажу криптовалюти за фіатні гроші. Це поширений нормативний підхід, і більшість криптовалютних бірж тепер повинні дотримуватися процедур «Знай свого клієнта» (KYC), які включають кілька кроків для ідентифікації користувача. Процедура KYC зазвичай включає кілька етапів, як-от надання ідентифікаційних документів і перевірка особистої

інформації користувача, щоб підтвердити особу користувача. Після успішної ідентифікації користувача за допомогою цього процесу його криптовалютні транзакції на біржі можна пов'язати з його справжньою особистістю, що полегшує відстеження будь-якої підозрілої діяльності або спроб відмивання грошей.

Коли підписана транзакція передається на інші вузли, то інформація про відправника (наприклад, його IP-адреса) може бути записана вузлами або проміжними мережевими пристроями. Передача будь-якої інформації на сторонній сервер може потенційно встановити зв'язок між користувачем і адресою. Наприклад, здійснення покупки на веб-сайті, пошук транзакції або перевірка балансу в блокчейн-провіднику (експлорері) [52], або використання програми гаманця, яка використовує аналітику.

Інший метод зіставлення об'єктів з адресами – це ведення запису інформації, пов'язаної з адресами блокчейну, яка була опублікована об'єктом або використання відкрито доступних баз даних. Одним із прикладів такої бази даних є список зловмисників [53] або база даних санкціонованих адрес, які використовуються під час аналізу. Також може бути корисним скрапінг соціальних мереж, оскільки користувачі часто публікують адреси своїх гаманців. Основний недолік цього підходу полягає в тому, що вам потрібно налаштувати складну інфраструктуру та заздалегідь зібрати багато даних, а особи, яка вас цікавить, може навіть не бути у цих даних.

### **1.2.2 Аналіз потоків транзакцій**

Під потоком транзакцій блокчейну розуміється послідовності транзакцій, які відбуваються в мережі блокчейн. Ці послідовності можуть варіюватися від малих до великих і мати складні структури, що містять розгалуження та з'єднання. Складні структури можуть містити цінну інформацію, яку не видно на перший погляд, тому для її вилучення слід застосовувати різні методи.

Іноді ці послідовності можуть навмисно мати складні структури. Злочинці часто приховують рух транзакцій, очікуючи, що слідчі втратять слід. Однак, використовуючи правильні підходи та інструменти, можна отримати суттєве

уявлення про потік і розкрити деталі, які в іншому випадку могли б залишитися прихованими. У наступних розділах ми розглянемо загальні підходи до аналізу потоків транзакцій блокчейну.

Дані можуть бути отримані вручну з блокчейну за допомогою блокчейн-провідників або подібних інструментів, які забезпечують зв'язок із вузлами блокчейну. Зазвичай це веб-інструменти [54], які дозволяють користувачам отримувати доступ і навігацію вмістом блокчейну. Вони мають графічний інтерфейс для вивчення та аналізу даних блокчейну, таких як записи транзакцій, адреси рахунків і баланси. Основною функцією дослідника блокчейнів є полегшення пошуку конкретних транзакцій, перевірка балансів гаманців і вивчення мережових показників. Хоча блокчейн-провідники корисні для базових випадків, вони не підходять для обробки складних випадків, що включають довгі ланцюги транзакцій.

Візуалізація графіків використовується для обробки складних послідовностей транзакцій. Для певних потреб можна використовувати різні типи графіків і представлень [55], [56], [57]. Однак у більшості випадків бажано представити потік у вигляді графа, в якому адреси представлені як вузли, а транзакції як спрямовані ребра (рис. 1.4). Проте цей формат дає можливість стежити за передачею активів у найбільш простий спосіб. Такий формат все ще може бути важко зрозуміти, і тому корисно групувати, фільтрувати, виділяти та приховувати окремі елементи, вилучати чи відокремлювати цінну інформацію від нерелевантних даних.

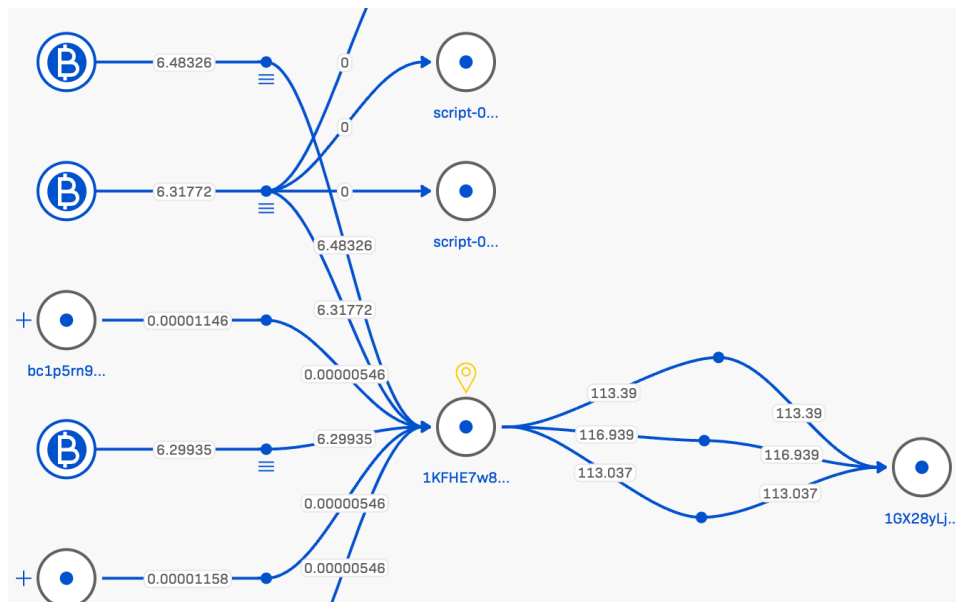


Рис. 1.4. Візуалізація графа в Crystal Explorer [58]

Користувачі можуть переводити свої активи з однієї мережі в іншу. Для цих цілей вони використовують біржові платформи та кросчейн-мости. Зловмисники, які хочуть обфускувати свій трафік, також можуть скористатися цими методами. Крім того, вони можуть використовувати такі інструменти, як міксери [59], [60], що може значно ускладнити завдання дослідників, які намагаються зрозуміти транзакційний потік. Ми можемо визначити ці інструменти як протоколи перетворення. Інструменти відстеження транзакцій мають бути обізнані про різні протоколи перетворення та бути достатньо надійними, щоб виконувати асоціювання адрес (у випадках, коли це теоретично можливо). У багатьох випадках протоколи перетворення, такі як кросчейн-мости [61], добре задокументовані, і якщо вони не призначені спеціально для приховування трафіку, вони можуть допомогти знайти адресу користувача в іншій мережі. У випадках, коли документація відсутня або якщо стверджується, що система забезпечує абсолютну анонімність, може знадобитися виконати ручний аналіз системи. Ретельний ручний аналіз системи може надати цінну інформацію про її функціональність. Це допомагає зрозуміти значення транзакцій користувача та виявити будь-які потенційні недоліки, якими можна скористатися. Крім того, цей тип аналізу може



відкрити можливості розробки нових методів отримання додаткової інформації про транзакції або користувачів [62].

Щоб спростити аналіз потоку, використовуються різні алгоритми для виявлення зв'язків, шаблонів і аномалій [63]. Їх можна використовувати для спрощення перегляду або виведення найважливіших даних на перший план. Ці алгоритми можуть бути класичними [64] або алгоритмами машинного навчання [65].

На ринку доступні комерційні інструменти, такі як Chainalysis Reactor, MistTrack та інші [66], які надають зручні інструменти для аналізу потоку, включаючи візуалізацію графіків та різні інші функції, які обговорюються в цьому розділі.

### **1.2.3 Аналіз смарт-контрактів**

Завдяки смарт-контрактам розробники та користувачі можуть задавати складні умови для переказу коштів, і як наслідок – будувати додаткові рівні абстракції та протоколи.

Ручний аналіз вихідного коду смарт-контракту дає повне уявлення про поведінку смарт-контракту. Цей процес передбачає ретельний аналіз коду, рядок за рядком, щоб отримати розуміння як загальної концепції, так і тонкощів. Однак цей тип аналізу вимагає глибоких знань мов програмування, криптографії та технології блокчейн, і може бути трудомістким процесом. Читання документації продукту може бути корисним і може прояснити причини деяких програмних рішень або пояснити незнайомі концепції, однак вона не завжди доступна.

Під час перевірки вихідного коду доступність вихідного коду є ще одним важливим аспектом, який слід враховувати. Вихідний код смарт-контракту зазвичай публікується, і в багатьох випадках його можна знайти на GitHub. Однак наявність вихідного коду контракту не означає, що той самий контракт опубліковано в блокчейні, тому необхідна перевірка контракту [67] – для відповідності вихідного коду та байт-коду на блокчейні.

У деяких випадках розробники можуть не робити вихідний код контрактів загальнодоступним. Як наслідок, необхідні альтернативні методи, щоб отримати розуміння поведінки контракту через аналіз його байт-коду. Загалом цей процес називається реверс-інжинірингом. Для спрощення процесу реверс-інжинірингу використовуються дизасемблери (перетворення байт-коду в коди операцій віртуальної машини Ethereum, Ethereum Virtual Machine – EVM) і декомпілятори [68], [69]. Декомпілятори перетворюють байт-код у представлення високого рівня. Однак, через втрату інформації під час компіляції, отриманий код не схожий на оригінальний код.

Якщо ми хочемо подивитися на фактичне виконання смарт-контракту на блокчейні, провідники можуть бути корисними для простих випадків. Деякі з них мають декодери транзакцій і можуть забезпечувати відстеження виконання коду. Існують інструменти, розроблені спеціально для декодування транзакцій, такі як Transaction Tracer [70] або подібні [71], які забезпечують трасування викликів, яке є деревом викликів функцій і аргументів, зроблених через різні контракти під час виконання транзакції. Крім того, існують інструменти для локального відстеження EVM [72], які дозволяють детально досліджувати транзакції смарт-контрактів. Середовища розробки, такі як Truffle, мають ще більш зручні засоби для налагодження транзакцій у блокчейні [73].

Інструменти автоматизованого аналізу смарт-контрактів можна використовувати для кращого розуміння смарт-контрактів. Зазвичай їх поділяють на дві категорії – статичні та динамічні аналізатори [74].

Інструменти статичного аналізу виконують аналіз контрактів без їх запуску. Slither framework [75] є одним із таких інструментів, призначений для автоматичного пошуку вразливостей, надання інформації про контракт та його функції, надання зведення про доступ до авторизації та багато іншого.

Інструменти динамічного аналізу, з іншого боку, виконують аналіз шляхом виконання смарт-контрактів або їх частин. Існують різні класи інструментів динамічного аналізу, що використовуються для аналізу смарт-контрактів, наприклад інструменти символічного виконання, SMT (Satisfiability Modulo

Theories) розв'язувачі, аналізатори “забруднень” (taint analysis) і фазери [76]. Mythril, Echidna [77] і Manticore [78] є одними з найбільш широко використовуваних інструментів для пошуку вразливостей у коді, пошуку набору вхідних даних, які переводять програму в неочікуваний стан, або для дослідження всіх можливих станів. Ці інструменти та підходи не є взаємовиключними, а скоріше вони дають різні погляди на те, як працює розумний контракт.

Комерційні інструменти, такі як MythX [79], поєднують статичний і динамічний підходи, щоб отримати найкраще з обох світів і забезпечити найповніші результати.

Останні дослідження та розробки штучного інтелекту (точніше, великих мовних моделей, таких як ChatGPT [80]), дозволили використовувати ці технології для пояснення коду, зворотного проектування [81] і навіть для пошуку вразливостей [82]. Ці інструменти використовуються вже зараз, а найближчим часом будуть ще більше використовуватись для допомоги під час аналізу коду.

### **1.3 Порівняльний аналіз сучасних систем аналізу блокчейн транзакцій**

На основі завдань, які повинні виконувати системи аналізу блокчейн транзакцій, можемо виділити основні критерії для порівняння існуючих системи аналізу:

- Пошук, фільтрування, сортування;
- Наявність графічного інтерфейсу та зображення графу транзакцій;
- Підтримка крос-мережевої взаємодії;
- Логічний аналіз транзакцій: розпізнавання логіки транзакцій до стандартизованих контрактів або розпізнавання скриптів в біткоїн-подібних мережах;
- Наявність бази даних з інформацією про облікові записи блокчейну.

#### **1.3.1 Блокчейн провідники**

Блокчейн провідники – засоби, що дозволяють користувачу отримати інформацію про блоки мережі, транзакції та іншу інформацію. Загалом, блокчейн-

провідник – це користувацький інтерфейс до блокчейн мережі. На рисунку 1.5 зображено приклад блокчейн-провідника Phalcon BlockSec. Хоча функціонал може відрізнятись від провідника до провідника основними функціями є:

- Отримання інформації про блокчейн мережу, кількість блоків, криптовалюти, хешрейт, кількість облікових записів;
- Надання інформації про блоки – поточний блок, складність, інформація про майнера та інші деталі.
- Перевірка транзакцій – користувач може отримати детальну інформацію про транзакції, кількість підтверджень, хто отримувач чи відправник;
- Баланс адреси або гаманця – дозволяє отримати поточний баланс криптовалюти та токенів, історії транзакцій вказаної адреси;
- Надання інформації про смарт-контракти та байт-код контракту.

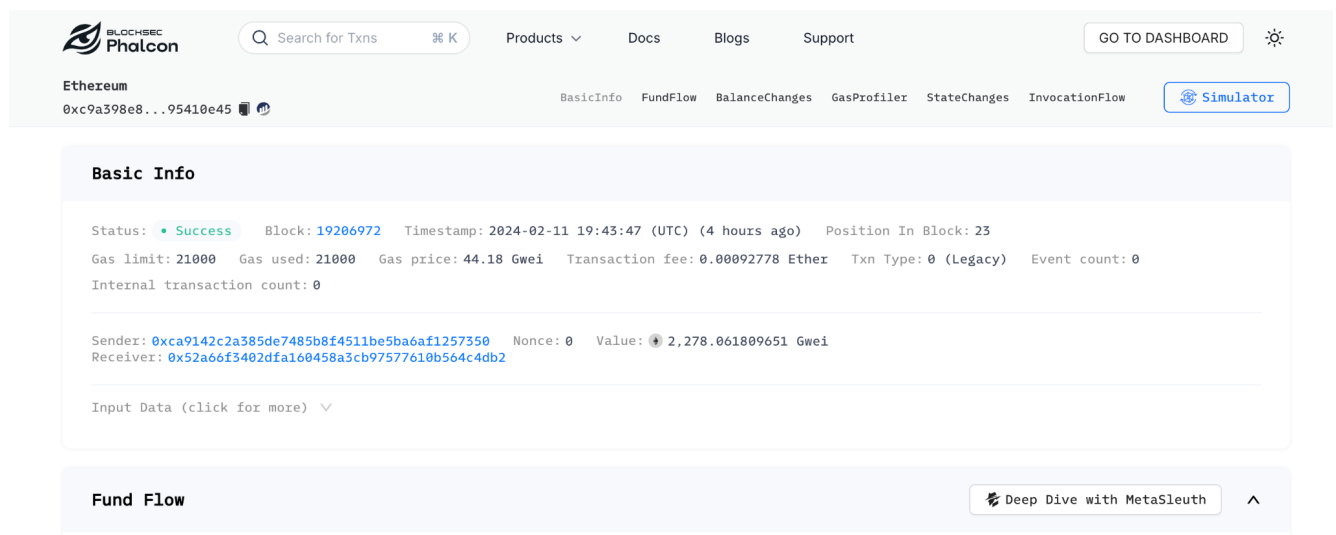


Рис. 1.5. Блокчейн-провідник Phalcon BlockSec

### 1.3.2 Комплексні системи дослідження блокчейн мереж

Складний аналіз, такий як виявлення незаконної діяльності, дослідження фінансових потоків, забезпечення дотримання правил боротьби з відмиванням грошей, потребує більш комплексних засобів. Дані засоби зазвичай мають просунутий графічний інтерфейс для візуалізації транзакцій, використовують різні

алгоритми, включаючи алгоритми машинного навчання для дослідження транзакцій, засоби аналізу потенційних ризиків та інші.

Однією з таких платформ є **Chainalysis** (рис. 1.6), яка використовується державними установами, біржами і фінансовими установами. Дана система допомагає відстежувати потоки коштів у блокчейн мережах, виявляти незаконні транзакції та аналізувати зв'язки між різними об'єктами.

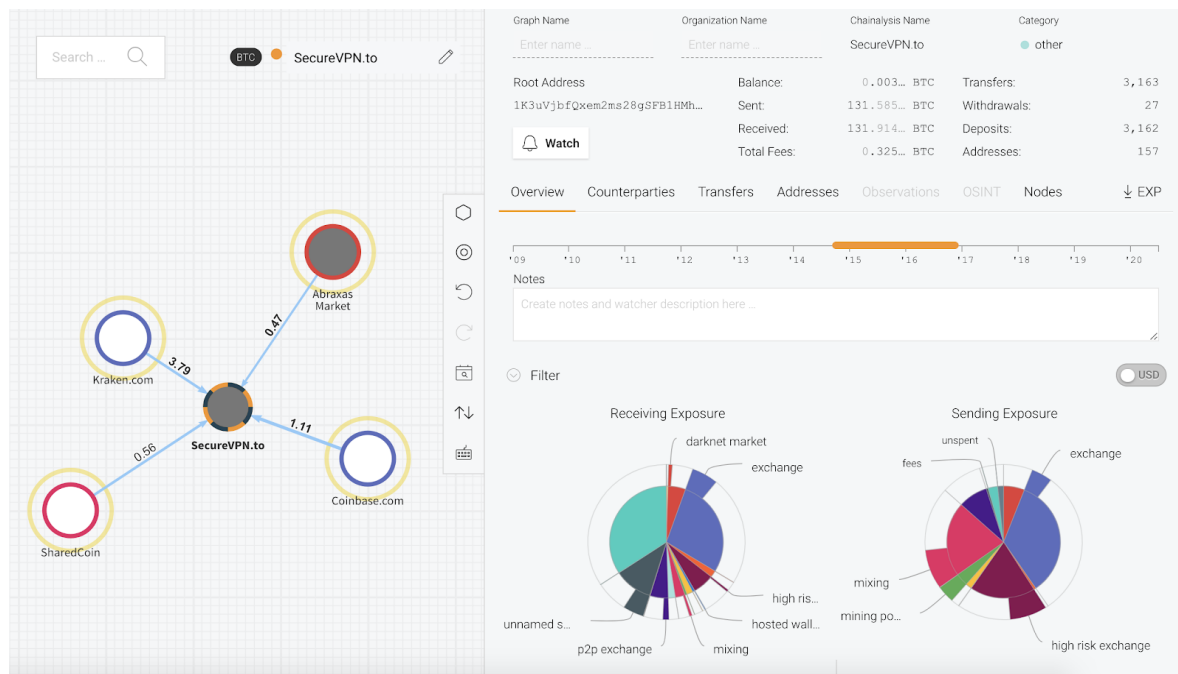


Рис. 1.6. Інтерфейс системи Chainalysis [83]

Іншою платформою дослідження блокчейн мереж є **Metasleuth**, яка може допомагати відстежувати потоки коштів злочинної діяльності, а також аналізувати кросчейн-взаємодії.

**MistTrack** – ще одна платформа, спрямована на надання інформації про потоки коштів у мережах блокчейнів, з акцентом на DeFi. Дана система допомагає виявляти шахрайство та відмивання грошей, а також розслідувати існуючі інциденти.

**Crystal Blockchain** – аналітична платформа, що надає інструменти для моніторингу та аналізу транзакцій. Crystal Blockchain підтримує кілька

криптовалют і надає рішення для судових розслідувань, оцінки ризиків, аналізу потоків транзакцій та інше.

**Arkham Intelligence** – це аналітична платформа, яка спеціалізується на моніторингу та аналізі блокчейн-транзакцій. Вона допомагає виявляти підозрілі транзакції, аналізувати ринки, та відстежувати переміщення між різними блокчейнами та криптовалютами (рис. 1.7).

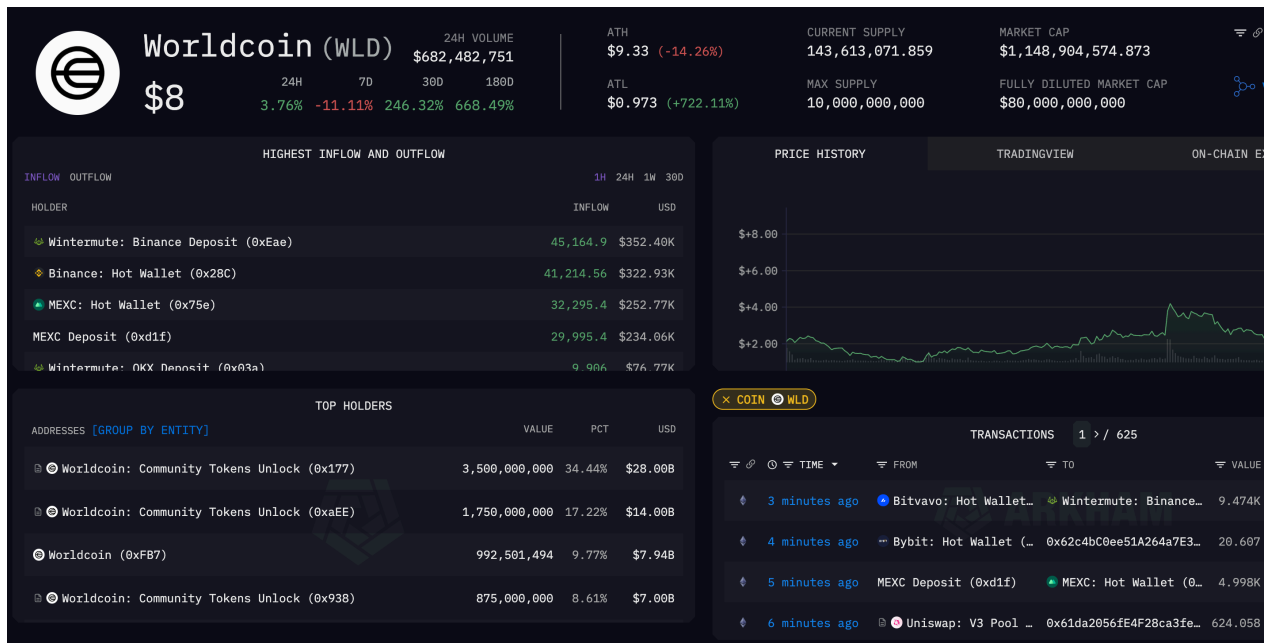


Рис. 1.7. Інтерфейс платформи Arkham Intelligence

### 1.3.3 Інші засоби дослідження

Крім комплексних систем аналізу наявні також і спеціалізовані інструменти для вирішення конкретних задач. Так, для прикладу, AMLBot [84] дозволяє перевіряти адреси та автоматизує AML/KYC процедури.

## 1.4 Огляд обмежень існуючих методів та постановка завдань дисертаційного дослідження

### 1.4.1 Збір даних з блокчейн-мереж для аналізу транзакцій

Аналіз блокчейн-транзакцій та смарт-контрактів в основному пов'язаний з дослідженням даних, збережених на блокчейні (ончейн-дані). Ці дані вимагають систематичного збору та організації для комплексного аналізу. Після розгортання

та синхронізації блокчейн-вузла можна використовувати інтерфейси, такі як JSON-RPC, для отримання необхідних даних [85]. У випадках, коли потрібен більш ретельний і обширний аналіз, віддається перевага прямому зв'язку з синхронізованою базою даних через можливі затримки, пов'язаних з використанням проміжних інтерфейсів.

У роботі [85] наводиться приклад аналізу блокчейн-транзакцій. Автори обрали блокчейн-мережу з невеликим об'ємом даних – PIVX. Розмір усіх даних у даній мережі займав лише 17 ГБ. Автори роботи завантажують дані з блокчейн-мережі в базу даних використовуючи JSON-RPC інтерфейс та ефективно проводять аналіз транзакцій у даній мережі. Але залишається невирішеними питанням, як масштабувати наведений підхід аналізу транзакцій для декількох мереж.

Зазвичай проведення блокчейн-аналізу передбачає налаштування особистого вузла. Цей підхід потребує наявності достатньо потужних обчислювальних систем. Для вузлів Ethereum рекомендується [86] мати мінімум 16 ГБ оперативної пам'яті, швидкість завантаження більше 25 Мбіт/с, тверdotілий накопичувач (SSD – solid-state drive) ємністю понад 650 ГБ для повної синхронізації вузла, та тверdotілим накопичувачем об'ємом 12 ТБ для архівного вузла. Крім того, процес синхронізації може зайняти від кількох днів до кількох тижнів. Проведення одночасного аналізу кількох мереж блокчейну вимагає значних зусиль та приносить з собою помітні витрати.

У роботі [87] описані способи доступу до блокчейн-мереж. Найбільшу увагу приділено трьом підходам: запуску локального повного вузла, виконання запитів до вузла як сервісу (node-as-service) та налаштуванні легкого вузла. У роботі подаються мінімальні системні вимоги для різних мереж та порівнюються різні node-as-service провайдери (кількість протоколів, які підтримуються провайдером, мінімальні витрати та інша інформація). Хоча описані методи виконують поставлену задачу отримання даних з блокчейн-мереж, вони вимагають розгортання вузла для кожної окремої мережі, що може бути неефективним при аналізі одразу декількох мереж.

В статті [88] автори презентують систему з відкритим вихідним кодом для блокчейн аналізу, з підтримкою різних мереж. Система передбачає використання розгорнутого повного блокчейн-вузла, з якої вона імпортує дані. Розроблена система є досить ефективною в частині обробки даних завантажених блокчейн-вузлом. Крім комунікації з вузлом через JSON-RPC інтерфейс, система має режим прямого парсингу бази даних без використання проміжних мережесих інтерфейсів. У будь-якому випадку, використання даної системи потребує розгортання повного блокчейн-вузла. Крім того, представлена система підтримує лише Bitcoin та похідні мережі.

У роботі [89] пропонується система для аналізу блокчейн транзакцій у вигляді блокчейн-провідника для мережі Bitcoin. Система використовує низку методів для аналізу, включаючи статистичні методи, кластеризацію відомих адрес, пошук шляхів та інші. Обмеженням даної роботи є те, що вона орієнтується лише на одну конкретну мережу та не передбачає аналізу декількох мереж.

У роботі [90] розроблено метод виявлення фінансової піраміди на основі аналізу байт-коду смарт контракту. Для отримання байт-коду смарт-контракту автори використовують блокчейн-провідник. Застосування блокчейн-провідника є виправданим у даному випадку, так як дозволяє заощадити ресурси та отримати байт-код контракту, не розгортаючи власного вузла. Хоча описаний метод виявлення фінансової піраміди теоретично може бути застосованим у багатьох блокчейн-мережах, робота не вказує як його масштабувати у даній мережі, тим паче, для інших мереж. Тому, обмеження цієї роботи полягає в частині отриманні необхідних даних. Блокчейн-провідник використовується лише як метод отримання інформації для конкретного смарт-контракту, а не масового аналізу контрактів багатьох блокчейн-мереж.

Загальний аналіз вищенаведених джерел дозволяє стверджувати, що нерозкритим питанням є питання масштабування, а основним недоліком описаних методів та рішень є їх орієнтація на конкретну блокчейн-мережу (або похідних однієї мережі), а також, в більшості випадків, використання розгорнутого персонального повного вузла конкретної мережі. При проведенні блокчейн аналізу



важливим є підтримка декількох блокчейн-мереж, так як типовим є ситуація, коли користувач конвертує криптовалюту однієї мережі в криптовалюту іншої, та продовжує свою діяльність в останній мережі. Масштабування оглянутих рішень в більшості є ресурсовитратним та займає значну частину часу в частині налаштування та синхронізації вузлів. Це дозволяє стверджувати, що доцільним є проведення дослідження, присвяченого розробці іншого методу для збору інформації з блокчейн мереж, який буде швидким та ресурсоекономним при аналізі декількох мереж.

#### **1.4.2 Деанонімізація користувача на мережевому рівні**

Зазвичай, будь-яка блокчейн-мережа використовує три типи мережевих протоколів для реалізації основного функціоналу [87] [91]: Node Discovery, Wire та JSON-RPC. Ці протоколи спільно керують складною мережевою взаємодією та комунікацій всередині блокчейн-екосистемах. Перший протокол, Node Discovery, сприяє ідентифікації та встановленню з'єднань між вузлами в мережі. У блокчейн-мережах цей протокол зазвичай реалізується за допомогою алгоритму Kademlia [92]. Протокол Wire, керує низькорівневою комунікацією між вузлами, визначаючи формат та структуру обмінюваних пакетів даних. JSON-RPC протокол надає можливість виклику віддалених процедур по HTTP протоколу, дозволяючи користувачам та стороннім застосункам отримувати інформацію з блокчейн-мережі та надсилати інформацію до неї. Ця тріада протоколів лежить в основі операційного фундаменту блокчейн-систем, організовуючи мережеву взаємодію.

Хоча в блокчейн-мережах зазвичай усі вузли вважаються рівноправними, на практиці це не зовсім так. В залежності від типу учасника мережі логування може виконуватись на різних рівнях та мати різну ефективність.

**Логування на звичайних вузлах.** Звичайні блокчейн-вузли є кістяком блокчейн-мереж. Дані вузли виконують завдання верифікації, яке полягає у впевненості в тому, що надіслані транзакції та блоки є правомірними та дотримуються встановлених протоколом правил. Після верифікації вузли ретранслюють транзакції та блоки до інших вузлів, а також додають ці блоки до

локальної копії ланцюга блоків (повної або часткової в залежності від типу вузла [93]). Це сприяє глобальному розповсюдженню та однорідності інформації, що забезпечує безпеку блокчейн-мереж.

Більшість вузлів працюють лише на низькорівневих протоколах Node Discovery і Wire та не надають публічний JSON-RPC інтерфейс. При використанні низькорівневих протоколів може здійснюватися логування, яке дозволить асоціювати IP-адресу вузла з надісланою блокчейн-інформацією.

Обмеженням логування на звичайному вузлі є неможливість визначити чи вузол, який надіслав транзакцію, створив її сам або лише ретранслював транзакцію створену іншим вузлом.

**Логування на публічних вузлах.** Ще одним учасником мережі, якого можна виділити, є публічні вузли, які надаються зацікавленими сторонами для публічного використання. Треті особи, такі як розробники електронних гаманців, бірж та інших децентралізованих застосунків, розгортають і пропонують свої публічні сервери для користувачів. Якщо в документації блокчейн-мережі вказаний певний публічний вузол для загального використання, то зазвичай фактичним власником даного вузла є саме розробник даної мережі.

Публічні вузли надають JSON-RPC інтерфейс для Web-, мобільних застосунків та для розширень браузера для комунікації з мережею.

Застосунки у своїх налаштуваннях зазвичай за замовчуванням використовують певний публічний вузол. На рис. 1.8 показаний приклад налаштування електронного гаманця Metamask за замовчуванням.

Оскільки користувач напряму під'єднується до вузла через JSON-RPC протокол, то логування, застосоване на рівні даного протоколу дозволяє у більшості випадків ідентифікувати цього користувача. Наприклад, асоціювати виклик процедури відправки транзакції з IP-адресою пристрою, з якого було здійснене підключення до сервера.

The image shows a configuration window for a blockchain network. It contains six labeled input fields, each with a default value:

- Network name:** Ethereum Mainnet
- New RPC URL:** <https://mainnet.infura.io/v3/>
- Chain ID:** 1
- Currency symbol:** ETH
- Block explorer URL (Optional):** <https://etherscan.io>

Рис. 1.8. Налаштування електронного гаманця Metamask за замовчуванням

Обмеженням даного методу є те, що застосунки за замовчуванням вказують публічний сервер, який має певну репутацію та довіру. Тому навіть, якщо розгорнути власний публічний сервер (або надати лише JSON-RPC інтерфейс), то навряд чи він буде застосований будь-яким децентралізованим додатком.

**Логування на вузлах завантаження.** Іншим особливим типом вузла є вузли завантаження (bootstrap/bootstrapping nodes). Вузол завантаження надає інформацію про конфігурацію мережі, в тому числі надає список вузлів до яких можливо здійснити підключення. Адреси вузлів завантажень прописані розробниками у налаштуваннях програмного забезпечення вузла, тому кожен новий клієнт мережі (вузол) спочатку підключається до вказаних вузлів завантаження, а вже потім до звичайних блокчейн-вузлів. При підключенні до будь-якого вузла (в тому числі вузла завантаження), вузол який підключається, отримує список інших вузлів та додається до списку доступних.

Логування на завантажувальних серверах дозволяє отримати список усіх IP-адрес вузлів, що підключаються до мережі. Крім того, Node Discovery протокол передбачає використання ідентифікатора вузла (згенерований на основі його публічного ключа), який може бути асоційованим з IP-адресою вузла користувача.

Вузли завантаження можуть бути розгорнуті будь-ким, але для того, щоб вони почали використовуватись як початкові вузли, вони повинні бути прописані у конфігураціях програмного забезпечення вузла. Для прикладу, у програмному забезпеченні вузлів Geth [94] за замовчуванням задані завантажувальні вузли [95], власниками яких є організація Ethereum Foundation [96].

Основним обмеженням застосування логування на завантажувальних серверах є те, що завантажувальні сервери суворо визначені у конфігураціях програмного забезпечення вузла, тому застосовувати даний метод можуть лише власники цих вузлів.

Ще одним обмеженням є те, що вузли завантаження не несуть основної логіки мережі і не обробляють транзакції. Тому не можна визначити з якого вузла, що підключався, була відправлена транзакція – доступні лише час підключення та IP-адреса цього вузла. Хоча це не може вважатись прямим методом ідентифікації кінцевого користувача, інформація про підключення до мережі і час підключення все ще може бути корисною в процесі аналізу. З іншого боку, якщо блокчейн-мережа досить мала, цієї інформації може бути достатньо для визначення IP-адреси користувача, який надіслав певну транзакцію.

Іншим недоліком даного методу є те, що він не гарантує, що вузол підключився до вузлів завантаження, які були вказані розробниками за замовчуванням. Користувач може самостійно вказати інші вузли для початкового підключення, тому до стандартних він може так і не підключитись.

**Логування на майнінг-вузлах.** Одним із особливих типів вузлів в Proof of Work (PoW) блокчейн-системах є майнінг-вузли (mining nodes). Ці вузли не лише беруть на себе завдання приймати, верифікувати та ретранслювати транзакції, але й здійснюють процес генерації нових блоків за допомогою майнінг-процесу [97]. Після успішного процесу майнінгу блока, блок ретранслюється до інших блокчейн-вузлів, а майнінг-вузол отримує певну кількість криптовалюти, як винагороду за успішне виконання майнінг-процесу.

Дані вузли використовують Node Discovery та Wire протоколи для комунікації з іншими вузлами. Також можуть мати активний JSON-RPC інтерфейс з додатковим функціоналом для управління процесом майнінгу [98].

Коли кінцевий користувач транслює створену транзакцію, вона ретранслюється іншими вузлами, доки ця транзакція не досягне майнінг-вузлів. Але кінцевий користувач може уникати проміжних вузлів та надсилати транзакцію напряму майнінг-вузлам, за умови, що він знає їхні адреси та не має обмежень на боці майнінг-вузлів. Це дозволяє уникнути аналізу та збору інформації про транзакцію користувача проміжними вузлами [99]. Таким чином, у разі прямого з'єднання, майнінг-вузол має можливість зв'язати транзакцію користувача з його IP-адресою.

Власниками майнінг вузлів можуть бути як приватні особи, так і великі корпорації. Немає обмежень на розгортання майнінг-вузлів і будь-хто може розгорнути майнінг вузол з невеликими ресурсами задля потенційного логування користувачів.

Основним обмеженням застосування логування на майнінг-вузлах задля ідентифікації кінцевого користувача є те, що клієнт повинен відправити транзакцію напряму до майнінг-вузла. В іншому випадку, логування на майнінг-вузлі не виправдане, адже транзакцію буде ретрансльовано проміжним вузлом і виникає така ж проблема як і при логування на звичайних вузлах.

**Логування у блокчейн-провідниках.** Іншим учасником блокчейн екосистем є блокчейн-провідники, які надаються більшістю блокчейн-мереж для своїх користувачів. Хоча блокчейн-провідники не є напряму елементами саме блокчейн-мереж, вони є важливим інструментом для її дослідження. Блокчейн-провідник – це зазвичай веб-застосунок, який дозволяє користувачам проводити аналіз та дослідження даних у блокчейн-мережі. Такими даними можуть бути транзакції, гаманці, блоки тощо. Він надає можливість детального вивчення та візуалізації даних, наприклад, для перевірки чи успішно була виконана транзакція користувача. Користувач зазвичай використовує блокчейн-провідник для

перевірки балансу свого аккаунту, або для пошуку відправленої транзакції та перевірки чи була вона включена у блокчейн.

Логування взаємодії користувача з блокчейн-провідником на рівні HTTP протоколу дозволяє достатньо точно встановити зв'язок IP-адреси користувача з інформацією, яку він запитував. Наприклад, якщо користувач запитує певну транзакцію, то це дозволяє асоціювати його IP-адресу з цією транзакцією, з відправником транзакції або отримувачем транзакції. Крім того, використання браузера для доступу до блокчейн-провідника збільшує ризики деанонізації користувача навіть у випадках використання VPN чи TOR [100] [101] [102].

Власниками блокчейн-провідників зазвичай є розробники блокчейн-мережі, приватні компанії та приватні особи [103], а розгортання власного провідника не гарантує його широке застосування. Це і є основним обмеженням.

Іншим обмеженням є те, що зв'язок IP-адреси з певною транзакцією чи аккаунтом не гарантує, що саме цей користувач є його власником або відправником транзакції. Проте це може спонукати на подальший аналіз цього зв'язка, наприклад визначити, чому даний користувач здійснив пошук у блокчейн-провіднику саме цієї транзакції.

#### **1.4.3 Виявлення вузлів блокчейн-мереж**

У статті [104] представлено NodeMaps, фреймворк для збору, аналізу та візуалізації даних з різних популярних блокчейн-платформ, таких як Cosmos, Stellar, Bitcoin, та Lightning Network. Автори порівнюють, як ці платформи розподілені по світу, які хостинги вони використовують і які програмні клієнти у них є. Вони виявили, що Bitcoin та Lightning Network мають велике географічне покриття та багато вузлів працюють через мережу TOR, яка забезпечує конфіденційність. Натомість, вузли блокчейн-мереж Cosmos та Stellar більше схильні працювати на великих хмарних платформах або в дата-центрах. Але даний метод не може бути застосований до інших мереж, так як протоколи пошуку вузлів відрізняються.

У статті [105] аналізується надійність P2P мережі Ethereum за допомогою розробленого інструмента для збору даних про вузли. Дослідження виявило сильну концентрацію вузлів в декількох автономних системах та підозрілі моделі, які можуть свідчити про атаку Sybil. Застосування цього методу виявлення до інших блокчейн-мереж, таких як Rootstock, є складним. Причиною цього є те, що краулер вимагає адаптації для відповідності унікальним механізмам та структурним особливостям кожної мережі.

Стаття [106] демонструє, що Go Ethereum (Geth), реалізація Ethereum, вразлива до атак типу Eclipse. Атака використовує логіку виявлення пірів Kademlia, яку використовує Geth, дозволяючи з легкістю ізолювати довготривалі віддалені вузли-жертви з використанням невеликих ресурсів. Автори обговорюють фундаментальні властивості логіки виявлення вузлів Geth, які дозволяють провести відповідну атаку. Вони аналізують протокол виявлення вузлів в мережі Ethereum, але результати даного аналізу не можуть бути прямо використані для інших блокчейн-мереж.

У статті [107] вивчається топологія мережі Bitcoin, в якій вузли розкидані по всьому світу. У роботі представлено BTCmap – фреймворк для виявлення та побудови карти мережі Bitcoin. Даний фреймворк включає два модулі: для збору баз даних адрес від кожного вузла та емулятор вузла Bitcoin для вибору сусідів та генерації топології. Застосування фреймворку BTCmap до інших блокчейн-мереж має потенціал, але водночас стикається з певними викликами. Кожна блокчейн-мережа має свої унікальні характеристики та архітектуру, що може вимагати адаптації та модифікації фреймворку для ефективного використання. Також важливо враховувати різні масштаби та складності мереж.

У статті [108] автори провели детальний збір даних про вузли Bitcoin протягом року, від липня 2021 до червня 2022, охопивши 127613 вузлів. Вони виявили, що кількість вузлів Bitcoin змінюється у зв'язку з коливаннями в мережі Onion. Автори також вибрали 2694 ключових вузлів Bitcoin і проаналізували їх важливі характеристики. В результаті вони використали машинне навчання для групування цих вузлів за різними атрибутами і зробили висновок про

централізацію в мережі Bitcoin. Дослідження вузлів Bitcoin має нерозв'язані питання, такі як аналіз недоступних вузлів та потреба в удосконаленні алгоритмів кластеризації та наборів даних. Ці виклики підкреслюють необхідність постійного вдосконалення методів виявлення вузлів для блокчейн-мереж, щоб краще розуміти їхню структуру та вразливості. Крім того, кожна блокчейн-мережа має свої специфічні характеристики, тому методи аналізу та виявлення можуть вимагати адаптації.

Стаття [109] описує новий метод TopoShot для виявлення топології блокчейн мережі. TopoShot можна використовувати з основними клієнтами Ethereum, такими як Geth та Parity. Одним із викликів даного дослідження є розширення методології для постійного моніторингу та аналізу динамічної топології основної мережі Ethereum в реальному часі, оскільки структура мережі та її зв'язки можуть швидко змінюватися. Крім того, застосування цього методу до інших блокчейн-мереж може потребувати значної адаптації через відмінності в мережевих протоколах та програмному забезпеченні клієнтів.

У роботі [110] досліджується топологія мережі Bitcoin, важлива для розповсюдження транзакцій та блоків. Автори розробили AddressProbe – метод, який виявляє зв'язки в P2P мережі Bitcoin. В дослідженні розглядаються повідомлення протоколу для отримання сусідніх вузлів. Аналіз топології дозволяє виявити вузли з високим ступенем з'єднання а також ідентифікувати впливові вузли, які з'єднані з майнінг-пулами. Серед невирішених проблем розробленого методу є проблема надійності. Метод покладається на неофіційні характеристики клієнта Bitcoin, які можуть бути змінені в майбутньому. Застосування методу AddressProbe до інших блокчейн-мереж може бути складним через різницю в їхній роботі, що вказує на потребу в більш гнучких методах для розуміння структури різних блокчейн-мереж.

Робота [111] присвячена вивченню P2P-мережі Ethereum. Автори аналізують мережеву структуру P2P системи Ethereum протягом двох місяців у головній мережі (mainnet), зібравши дані з понад 217514 різних вузлів. Зібрані дані дозволили провести аналіз статусу вузлів, типу сервісів вузлів, географічного



розподілу вузлів та провайдерів інтернет-послуг. Дослідження також виділяє декілька невирішених питань, зокрема потребу у більш детальному аналізі поширення інформації, що може бути покращено за допомогою збору більшої кількості даних та збільшення кількості мережових з'єднань. Крім того, існує потенціал для розробки критеріїв включення транзакцій майнерами. Застосування цих методів виявлення до інших блокчейн-мереж вимагатиме адаптації до унікальних характеристик кожної мережі, що підкреслює потребу у використанні індивідуалізованих підходів у аналізі блокчейн мереж.

У роботі [112] автори пропонують метод Ethna, який вимірює ступені вузлів Ethereum та визначає затримку при розповсюдженні повідомлень в P2P мережі Ethereum. Також у роботі проведені експерименти для аналізу топологічних особливостей відповідної P2P мережі. Однак поза увагою залишаються питання, такі як необхідність постійного аналізу для відстеження еволюції мережі. Крім того, методи виявлення вузлів, розроблені для мережі Ethereum, не можуть бути прямо застосовні з іншими P2P системами.

#### **1.4.4 Ідентифікація відправника транзакцій**

У роботі [113] автори досліджують стан анонімності блокчейн-мереж на мережевому рівні. Автори показують, що вузол, що має достатньо зв'язків з іншими вузлами може використати час відправлення транзакцій для розкриття інформації про їх походження у мережі Bitcoin. Описаний підхід не може бути застосований до інших блокчейн-мереж без попереднього аналізу їх мережових протоколів та відповідних адаптацій.

У роботі [114] аналізується стан анонімності Bitcoin на мережевому рівні, автори моделюють мережовий стек Bitcoin та досліджують його властивості. Автори показують, що мережеві протоколи Bitcoin (як до, так і після 2015 року) мають слабкі властивості анонімності.

У роботі [115] автори пропонують метод деанонімізації відправника транзакцій, шляхом пасивного прослуховування транзакцій без встановлення з'єднань з вузлами. Описане підхід має суттєвий недолік – він потребує доступу до

інтернет-інфраструктури, зокрема до автономних систем (Autonomous System) та точок обміну Інтернет-трафіком (Internet Exchange Point). Крім того, застосування даного підходу потребує адаптації до інших блокчейн-мереж.

У роботі [116] пропонується метод деанонізації Bitcoin-вузлів, які недоступні для вхідних з'єднань і знаходяться за маршрутизатором. Метод базується на аналізі запитів блоків, які вузол робить під час сесій в мережі Bitcoin, та організації цих запитів у граф сесій. Ідея полягає в тому, що вузол, який запитує блоки, робить це в порядку зростання їх висоти, що дозволяє зв'язувати послідовні сесії даного вузла. Проте, даний метод не може бути застосований саме для визначення відправника транзакцій, а лише для розрізнення вузлів що мають однакову IP-адресу (знаходяться за маршрутизатором).

#### **1.4.5 Bitcoin скрипти та OP\_RETURN опкод**

Як зазначено у розділі 1.1 мережа Bitcoin використовує поняття скриптів блокування (scriptPubKey) та розблокування (scriptSig), як альтернативу заданню отримувача транзакції. Скрипти задаються опкодами і одним із таких опкодів є OP\_RETURN. Він є особливим, адже застосовується для розміщення довільних даних на блокчейні. Розміщення даних дозволяє будувати протоколи із застосуванням блокчейну.

Аналіз Bitcoin транзакцій передбачає розпізнавання різних типів Bitcoin скриптів [117], в тому числі OP\_RETURN-скрипти (також називають nulldata-скриптами). Існуючі дослідження розглядають різні типи скриптів [118] [119], включаючи OP\_RETURN-скрипти [120] [121] [122] та нестандартні (non-standard script) [123] [124].

У роботі [120] представлено метод аналізу OP\_RETURN-транзакцій, що включає три кроки: пошук в інтернеті відомих асоціацій між ідентифікаторами та протоколами; класифікація OP\_RETURN-скриптів, що починаються з отриманих ідентифікаторів; виконання частотного аналізу перших кількох байтів метаданих на некласифікованих OP\_RETURN-скриптах для виявлення нових ідентифікаторів протоколів. В результаті першого кроку автори виявили 22 протоколи з 33

ідентифікаторами; на другому кроці було асоційовано 970374 транзакції з цими протоколами (~51% усіх OP\_RETURN), класифікуючи решту як порожні (~15%) або невідомі (~32%); останній крок, що передбачає автоматизований аналіз не виявив значущих статистичних аномалій та не привів до виявлення нових ідентифікаторів.

У роботі [122] проводиться систематичний аналіз використання метаданих у блокчейні Bitcoin. Автори обговорюють різні техніки вбудовування метаданих у блокчейн, серед яких OP\_RETURN скрипти є найпоширенішим способом.

#### 1.4.4 Постановка завдань

Враховуючи найбільшу капіталізацію Bitcoin серед усіх блокчейн-мереж [125] та популярність її сайдчейну Rootstock [126], який надає можливість використання смарт-контрактів, основний акцент у роботі робиться саме на цих мережах. Табл. 1.1 відображає результати порівнянь існуючих підходів та їх обмежень для вирішенні задач аналізу транзакцій.

Таблиця 1.1. Результати порівнянь існуючих методів

Задача	Короткий опис підходу	Обмеження та недоліки
Збір блокчейн-даних	Розгортання та синхронізація власних вузлів	Вимагає значних ресурсів. Не надає офчейн дані.
Збір блокчейн-даних	Використання JSON-RPC інтерфейсу на публічних вузлах	Не надає офчейн дані (вихідні коди смарт-контрактів). Інтерфейс може мати обмеження на надання даних.
Виявлення вузлів мережі	Послідовне опитування вузлів на надання сусідів	Не розроблено для мережі Rootstock
Виявлення відправника транзакції	Підключення до вузлів та аналіз часу отримання транзакцій на мережевому рівні	Не розроблено для мережі Rootstock
Аналіз OP_RETURN-скриптів	Попередній пошук відомих ідентифікаторів [120], [122]	Вимагає пошуку відомих асоціацій між

		ідентифікаторами та протоколами.
Аналіз OP_RETURN-скриптів	Аналіз частоти появи префіксів для пошуку аномалій [120]	Не виявив значні аномалії, і як наслідок нові префікси

Поточний стан області аналізу блокчейнів та обмежень існуючих методів дозволяє сформулювати наступні задачі для надання ширших можливостей при аналізі блокчейн систем та транзакцій:

- Розробити метод та програмне забезпечення збору даних з різних блокчейн мереж;
- Розробити метод та програмне забезпечення виявлення вузлів блокчейн мережі Rootstock;
- Розробити метод та програмне забезпечення ідентифікації відправника транзакцій блокчейн мережі Rootstock;
- Розробити метод та програмне забезпечення аналізу OP\_RETURN-скриптів блокчейн мережі Bitcoin;
- Запропонувати архітектуру системи аналізу блокчейн транзакцій та інтеграцію розроблених методів до такої системи.

На рис 1.9 зображено групи задач, що вирішуються даними методи для розробки.



Рис. 1.9. Методи для розроблення, призначені для вирішення відповідних груп задач

### 1.5 Висновки до розділу 1

У даному розділі проведений аналіз засобів та методів дослідження блокчейн транзакцій. Розглянуто області застосування такого аналізу у різних сферах. Виділено три основні групи задач, які він намагається вирішити: асоціювання адреси з особою, аналіз великих потоків транзакцій та аналіз смарт-контрактів. Розглянуто методи вирішення даних задач. Проведено порівняльний аналіз сучасних систем аналізу блокчейн транзакцій, серед яких є блокчейн-провідники та комплексні платформи дослідження транзакцій. Проведено огляд існуючих методів аналізу та їх обмежень, які дане дисертаційне дослідження вирішує.

## **РОЗДІЛ 2 АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ МЕТОДІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ**

### **2.1 Метод застосування блокчейн-провідників для збору даних з різних мереж**

#### **2.1.1 Основні етапи методу**

Метод застосування блокчейн-провідників для отримання даних для аналізу блокчейн-мереж включає наступні дії.

Крок 1. Вибір блокчейн-провідника. Вибір провідника є першим і основним кроком та впливає на функціональність методу. Під час вибору провідника важливо врахувати задачу, яку вирішує поточний блокчейн аналіз, адже це дає зрозуміти, яка саме інформація буде збиратись та з якою інтенсивністю, що дозволить виявити потенційні обмеження та доцільність використання методу. Серед критеріїв, які можуть вплинути на вибір провідника є:

- інформація, що надається провідником. Блокчейн-провідник – це проміжний інтерфейс між блокчейн-мережею та кінцевим користувачем, тому дані, які він надає кінцевому користувачу не повністю ідентичні тим, які зберігається на блокчейні, а піддаються попередній обробці перед відправкою їх до користувача. Це не є обов'язково негативним фактором, адже під час обробки даних корисна інформація виокремлюється та ефективно агрегується, а також може кешуватись, що прискорює загальний процес. Крім того, блокчейн-провідники можуть надавати додаткову інформацію, яка не зберігається на блокчейні, наприклад, вихідні коди верифікованих (verified) смарт-контрактів;

- наявність публічних серверів провідника для обраних мереж у яких проводиться аналіз;

- швидкість комунікації з серверами, на яких розгорнуті блокчейн-провідники та обмеження на кількість запитів;

- відкритий вихідний код провідника. Відкритий вихідний код провідника дозволяє краще розуміти його внутрішні механізми, і, як наслідок, дозволяє виявляти потенційно неефективні частини та обмеження. Крім цього, вихідний код

спрощує програмну реалізацію запропонованого методу. Також він дозволяє інтегрувати нові блокчейн-мережі, які на даний момент не підтримуються блокчейн-провідником.

Крок 2. Аналіз архітектури. Проводиться аналіз архітектури існуючої або нової системи аналізу блокчейн-мереж, в яку буде інтегрована програмна реалізація методу. Важливою частиною аналізу архітектури є визначення можливостей для масштабування і паралелізації процесів для одночасного збору даних з багатьох блокчейн-мереж.

Крок 3. Визначення способу комунікації з блокчейн-провідником. Визначається як саме буде отримуватись інформація з провідника: використовуючи прикладний програмний інтерфейс (application programming interface – API) чи звичайний веб-інтерфейс (що передбачає парсинг веб-сторінок). Також визначаються кінцеві точки (ендпоїнти – endpoints) блокчейн-провідника, які будуть використовуватись для отримання потрібних даних.

Крок 4. Програмна реалізація збору даних. На даному етапі аналізуються програмні засоби та виконується розробка відповідних модулів для комунікації з блокчейн-провідником для отримання даних на основі розробленої архітектури з урахуванням паралелізації.

### **2.1.2 Вибір блокчейн-провідника**

Блокчейн-провідники представляють собою веб-інструменти, призначені для полегшення дослідження блокчейн мереж. Ці інструменти дозволяють користувачам ефективно шукати, фільтрувати та сортувати транзакції, блоки, облікові записи та іншу інформацію в блокчейн-екосистемі. Вони мають широкий спектр застосувань, що охоплює прості завдання, такі як перевірка існування транзакції переказу токенів, а також набагато складніші, як дослідження певної злочинної діяльності. Хоча функціональність цих інструментів дуже схожа на те, що надає інтерфейс JSON-RPC (головним чином тому, що провідники використовують цей інтерфейс), вони збирають і надають додаткову інформацію.

Завдяки блокчейн-провідникам користувачі отримують доступ до ряду функціональностей, що значно спрощують процес взаємодії з даними блокчейну. Для базових випадків використання, блокчейн-провідники дозволяють користувачам швидко перевіряти транзакції, перевіряти баланси та відстежувати підтвердження певних транзакцій або блоків. Більш складні провідники надають список методів смарт-контрактів і дозволяють взаємодіяти зі смарт-контрактами. Більше того, деякі з провідників надають інформацію, таку як верифіковані смарт-контракти [127], що означає доступ до вихідного коду, пов'язаного з розгорнутим байт-кодом.

У даному дослідженні акцент зосереджений на сумісних з EVM провідниках. Також надається перевага провіднику з відкритим вихідним кодом, оскільки для нових блокчейн-мереж їх легше інтегрувати, аніж створювати власний провідник. В табл. 2.1 наведене порівняння різних блокчейн-провідників за ключовими характеристикам для вибору того, який найкраще підходить для застосування методу.

Таблиця 2.1. Порівняння різних рішень блокчейн-провідників

Назва провідника	Відкритий вихідний код	Підтримувані мережі	Байт-код смарт контрактів	Верифікований вихідний код смарт контракту
Blockchair [128]	Ні	17	Ні	Ні
Blockscout [129]	Так	200+	Так	Так
Blockchain [130]	Ні	3	Ні	Ні
Bitquery [131]	Так	50+	Ні	Ні
Etherscan [132]	Ні	19	Так	Так
Unmarshal Xscan [133]	Ні	18	Ні	Ні
Ethereum Beacon Chain [134]	Так	4+	Ні	Ні
Otterscan [135]	Ні	–	Ні	Так



Blockhead [136]	Так	—	Hi	Hi
Eternal [137]	Так	—	Hi	Hi
3xpl [138]	Hi	20	Hi	Hi
EthVM [139]	Так	2	Hi	Hi
BlockCypher [140]	Так	7	Hi	Hi
BlockExplorer.one [141]	Hi	10	Hi	Hi
Tokenview [142]	Hi	120+	Так	Так
CoinMarketCap [143]	Hi	2	Hi	Hi

Провідники зазвичай мають схожий інтерфейс. На рис. 2.1 зображена головна сторінка провідника Blockscout.

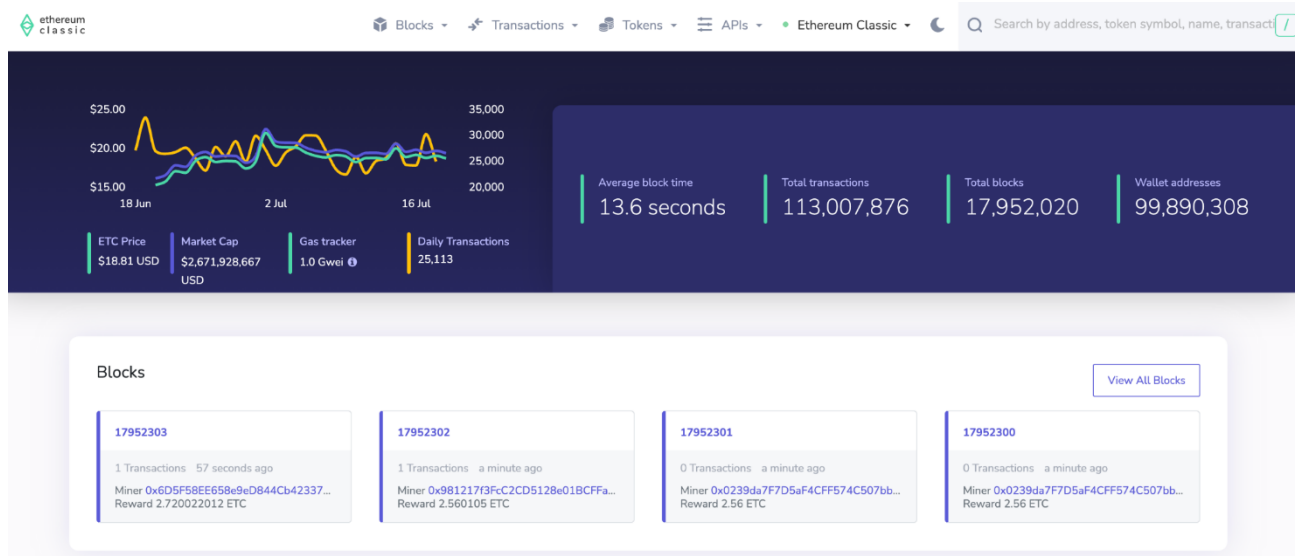


Рис. 2.1. Головна сторінка провідника Blockscout

Аналізуючи різні блокчейн-мережі та провідники було дійдено висновку, що Blockscout є найкращим кандидатом для подальших дослідницьких цілей, хоча для інших завдань він може бути не оптимальним.

### 2.1.3 Схема збору даних для аналізу блокчейн мереж

Дане дослідження зосереджене на швидкому отриманні даних одночасно з багатьох блокчейн мереж. Загальна структура системи досить проста і показана на рис. 2.2. Вона передбачає послідовності функцій отримання та обробки даних.

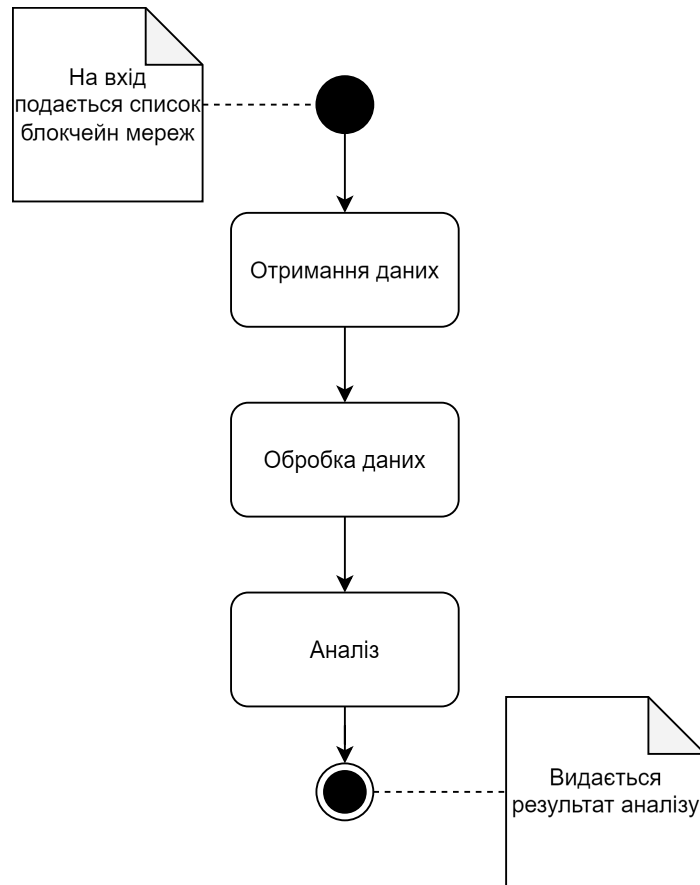


Рис. 2.2. Загальна схема отримання та обробки даних

Загальний підхід працює наступним чином. На вхід подається список блокчейн мереж (адреси блокчейн-провідників). Потім дані отримуються з провідників і обробляються. На основі оброблених даних та визначеної логіки отримується додаткова інформація, яка також піддається обробці. На кінцевому етапі отримані дані та інформація подаються для подальшого аналізу (ручного або автоматизованого).

У даному дослідженні схема має більш складну структуру (рис. 2.3) і зосереджена на зборі інформації про наявні смарт-контракти у блокчейн-мережах. Після кожного етапу отримання відповідь оброблюється, щоб виділити необхідну інформацію та зберегти її у зручному форматі для офлайн доступу. Спочатку отримується інформація про облікові записи, потім фільтруються смарт-контракти, отримується інформація про наявні токени, витягується вихідний код контрактів, якщо він доступний, та декомпілюється розгорнутий байт-код.

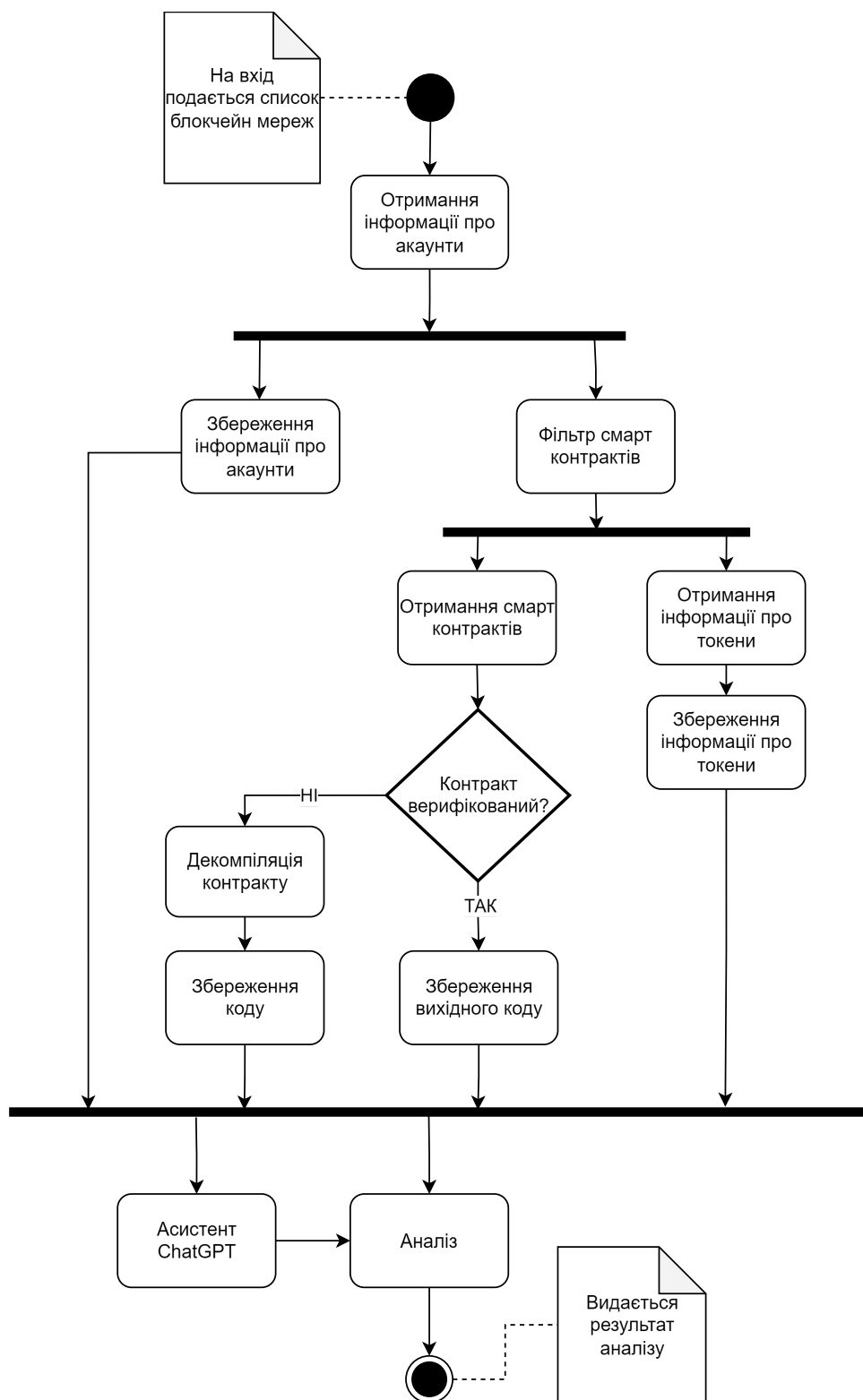


Рис. 2.3. Схема збору та обробки даних для запропонованого рішення

Під час фази аналізу обрані вихідні коди смарт-контрактів додатково попередньо аналізуються за допомогою системи ChatGPT [144], щоб дати короткий огляд, який може бути корисним під час масштабного аналізу. Для цієї цілі використовується API системи ChatGPT [145]. Хоча для цієї мети можуть

використовуватись інші великі мовні моделі (Large Language Model – LLM), детальний аналіз, порівняння та вибір їх виходить за межі даного дослідження. Тут він служить для демонстрації можливого компонента системи масштабованого аналізу.

У даному дослідженні використовується модель gpt-3.5-turbo, яка є однією з версій моделі GPT-3.5, відрізняється своєю оптимізацією для швидкої роботи. Моделі є недетермінованими, що означає, що вони надають різні результати для одних і тих самих вхідних даних. Значення параметра temperature (від 0 до 2) вказує на ступінь випадковості результату. У цьому дослідженні використовується значення 0.3. Щодо запиту, який надається ChatGPT, то він має таку структуру: «Give a brief summary of the next Smart Contract:» та додається вихідний код смарт-контракту.

#### 2.1.4 Формалізоване представлення методу

Процес збору даних представлений :

1. Для кожного блокчейн-провідника  $u \in U$  виконати запит до кожної сторінки  $p \in P$  для отримання її вмісту  $s_{up}$ .
2. Застосувати функції парсингу  $t_p \in T$  до вмісту веб-сторінки  $s_{up}$ .
3. Результатом є дані  $d_{up} \in D$  виокремлені з кожної сторінки.

Модель:

$$F: (U, P, T) \rightarrow D$$

де

$F$  – функція збору даних.

$U$  – множина URL-адрес провідників (Blockscout) для обраних блокчейн-мереж.

$P$  – множина шляхів до веб-сторінок обраного типу провідника, що містять корисну інформацію.  $P = \{"/accounts", "/address/{address}/contracts", "/address/{address}/token - balances"\}$ . Детальний опис шляхів наведено у розділі 3.3.

$T$  – множина функцій парсингу для кожного заданого шляху  $P$ . Функції  $t_p \in T$  обробляють HTML-теги отриманих сторінок. Функції наведені у розділі 3.3.

$D$  – отримані дані з кожної сторінки (інформація про облікові записи, смарт-контракти, баланси токенів).

## 2.2 Метод виявлення та ідентифікації блокчейн вузлів

### 2.2.1 Особливості Peer Discovery протоколу

Peer Discovery – загальна назва протоколів для пошуку вузлів у P2P мережах, які часто реалізуються за допомогою розподіленої хеш-таблиці Kademlia (Distributed Hash Table) [146] на UDP протоколі. Rootstock мережа не є виключенням. Основна логіка протоколу реалізована у класі PeerExplorer [147]. Приклад аналізу показано на рис. 2.4.



Рис. 2.4. Аналіз вихідного коду реалізації протоколу

У класі PeerExplorer відбувається обробка повідомлень та робота з іншими частинами протоколу, наприклад, розподіленою таблицею вузлів.

**Розподілена хеш-таблиця.** Kademlia DHT реалізується у класі NodeDistanceTable [148] та містить слоти з вузлами відповідно до їх відстані до поточного вузла. Параметри таблиці передбачають [149]:

– 256 слотів;

– 16 вузлів у слоті.

Важливим елементом алгоритму є ідентифікатор вузла (*NodeId*). Ідентифікатор RSK вузла – це публічний ключ вузла без першого байту формату [150]. Відстань між вузлами розраховується на основі цього ідентифікатора, а точніше, на основі хешу ідентифікатора  $keccak256(keccak256(NodeId))$ . Відстань – це позиція старшого біта (Most Significant Bit) значення операції XOR над хешованими значеннями ідентифікаторів двох вузлів (рис. 2.5).

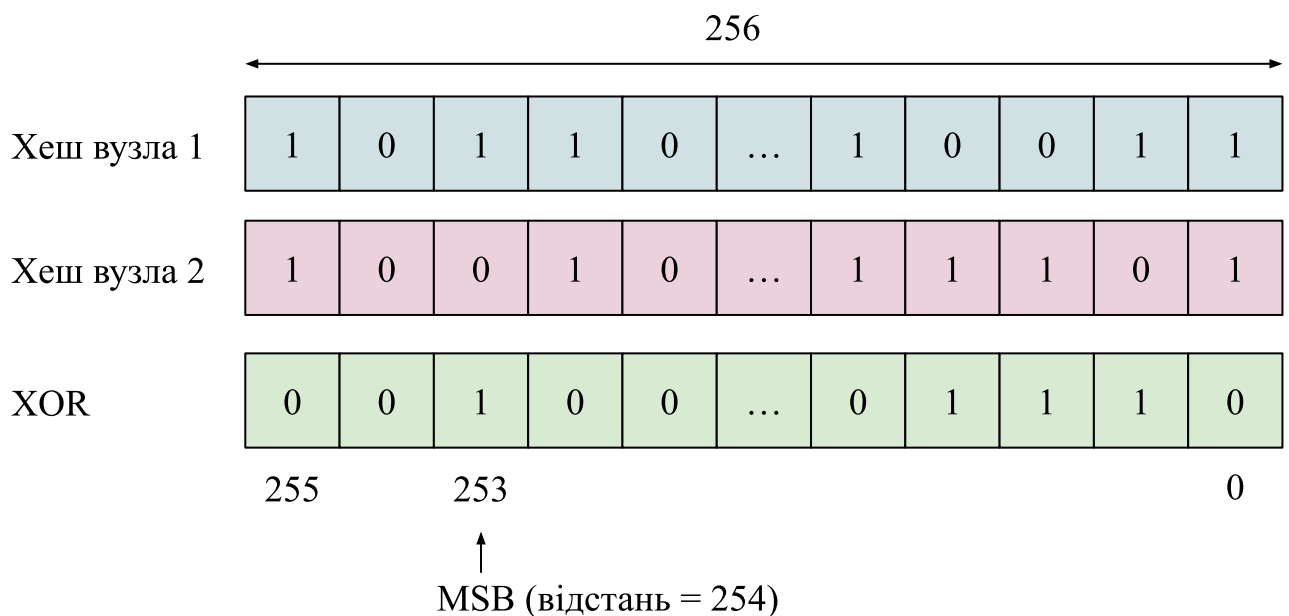


Рис. 2.5. Розрахунок відстані між вузлами

Для слота з відстанню 1, хеш ідентифікаторів вузлів є майже однаковим і відрізнятися лише одним молодшим (нульовим) бітом. На практиці це означає, що у слоти більшої відстані буде потрапляти більше вузлів ніж у слоти меншої відстані.

**Формат повідомлень.** Протокол реалізується за допомогою чотирьох типів RLP повідомлень [151]: два типи запитів і два типи відповідей. Перед відправкою RLP повідомлення обертаються додатковою інформацією. Загалом пакет, який надсилається через UDP-протокол, має структуру, зображену на рис. 2.6.

MDC (32 байта)	Signature (65 байтів)	Type (1 байт)	Data (N байтів)
-------------------	--------------------------	------------------	--------------------

Рис. 2.6. Формат повідомлення

Поля мають наступні значення:

- Data – RLP повідомлення;
- Type – тип RLP повідомлення (PING, PONG, FIND\_NODE чи NEIGHBORS);
- Signature – ECDSA підпис даних (полів Type та Data), заданий параметрами  $v$  (32 байти),  $r$  (32 байти) та  $s$  (1 байт). Підпис здійснюється за допомогою приватного ключа вузла;
- MDC – контрольна сума, яка формується на основі решти даних, а саме,  $\text{keccak256}(\text{Signature} + \text{Type} + \text{Data})$ .

В усіх RLP повідомленнях наявні 2 обов’язкові поля `rlpCheck` та `rlpNetworkId`. `rlpCheck` – унікальний ідентифікатор (Universally Unique Identifier – UUID), використовується для перевірки, що відповідь прийшла на вказаний запит, тобто відповідь повинна містити те саме значення `rlpCheck`, що й запит. `rlpNetworkId` – ідентифікатор мережі, у RSK мережі можливі значення 775 та 8100 для `mainnet` та `testnet` відповідно.

Формат 4 типів повідомлень наведений нижче:

- PING. Запит на відповідь, використовується для перевірки доступності вузла та для ініціалізації з’єднання. Після отримання даного повідомлення вузол-одержувач надсилає PONG-відповідь, а також PING-запит, якщо з’єднання між вузлами не було ініціалізоване до цього.

Хоча повідомлення цього типу у інших блокчейн-мережах [152] передбачає зазначення окремо UDP (для даного протоколу) та TCP (для Wire протоколу) портів, у RSKj використовується один і той самий порт:

$[[rlpIp, rlpPort, rlpPort], [rlpIpTo, rlpPortTo, rlpPortTo], rlpCheck, rlpNetworkID]$ .

Крім того, формат повідомлення передбачає зазначення параметрів з'єднання вузла-відправника та вузла-отримувача, RSKj зазначає параметри вузла-відправника замість вузла-отримувача, тому остаточний формат повідомлення має наступний вигляд:

$[[rlpIp, rlpPort, rlpPort], [rlpIp, rlpPort, rlpPort], rlpCheck, rlpNetworkID]$ ,

де *rlpIp* – це IP-адреса відправника даного повідомлення, *rlpPort* – порт підключення.

– PONG. Відповідь на PING повідомлення. При отриманні PONG відповіді вузол-отримувач намагається додати вузла-відправника до Kademlia таблиці. Додавання до таблиці означає, що з'єднання між вузлами встановлене. Якщо слот для поточної відстані заповнений, то додавання до таблиці не відбувається та з'єднання не вважається встановленим.

Як і в PING повідомленні, у повідомленні дублюється параметри з'єднання відправника повідомлення та використовується один порт для UDP та TCP протоколів:

$[[rlpIp, rlpPort, rlpPort], [rlpIp, rlpPort, rlpPort], rlpCheck, rlpNetworkID]$ ,

де *rlpIp* – це IP-адреса відправника даного повідомлення, *rlpPort* – порт підключення.

– FIND\_NODE. Запит сусідніх вузлів. Повідомлення має наступний формат:  $[rlpNodeId, rlpCheck, rlpNetworkId]$ . Де *rlpNodeId* – ідентифікатор вузла (на основі публічного ключа вузла), для якого будуть надані сусідні вузли.

– NEIGHBORS. Відповідь на FIND\_NODE повідомлення. Одне повідомлення надає до 20 сусідніх вузлів (MAX\_NODES\_PER\_MSG) з Kademlia таблиці за одне



повідомлення. Серед них 15 найближчих сусідів [153] та 5 випадкових [154]. Під найближчими сусідами розуміються сусіди, логічна відстань між якими та між *rlpNodeId* (вказаний запитом) є найменшою. Відповідь надається лише, якщо вузол вважається підключеним, тобто сам знаходиться у розподіленій Kademlia таблиці.

Відповідь має наступний формат: *[[node1, node2, node2, ..., node20], rlpCheck, rlpNetworkId]*, де *node* це сусідній вузол заданий наступним списком *[rlpHost, rlpUDPPort, rlpTCPPort, rlpId]*. *rlpHost* – IP-адреса або ім'я хоста вузла, *rlpUDPPort* – UDP порт підключення, *rlpTCPPort* – TCP порт підключення, *rlpId* – ідентифікатор вузла заснований на публічному ключі вузла.

### 2.2.2 Основні етапи методу

Метод передбачає наступні кроки:

Крок 1. Створити порожню чергу для відстеження вузлів, у яких потрібно запитати сусідні вузли, та список для відстеження всіх отриманих вузлів.

Крок 2. Визначити початкові вузли для пошуку, які зазвичай задаються у конфігураційному файлі [155] або як вхідні дані. Додати ці початкові вузли до черги на опитування.

Крок 3. Дістати вузол з черги для подальшої обробки. Обробка виконується у наступних кроках.

Крок 4. Встановити з'єднання з обраним вузлом. Для встановлення з'єднання потрібно: надіслати PING, отримати PONG, отримати PING, надіслати PONG.

Якщо у вузла, до якого здійснюється підключення заповнений увесь слот у розподіленій хеш-таблиці для заданої відстані, то при обробці PONG-відповіді він не додасть нового вузла до списку підключень. А отже, не буде надавати відповіді новому вузлу на запити надати сусідів.

Даний етап можливо оптимізувати: генерувати приватний ключ вузла, доки не буде отриманий публічний ключ, який дозволить отримати меншу відстань *K* до вузла, до якого здійснюється підключення. Завдяки цьому вузол буде додавати наш вузол до слота ближчої відстані, тобто менш заповненого слота. Це збільшить шанси встановити з'єднання з першої спроби;

Крок 5. Запитати сусідні вузли у підключеного вузла. Надіслати запит FIND\_NODE та отримати відповідь NEIGHBORS.

Крок 6. Додати сусідні вузли, які не знаходяться у списку всіх отриманих вузлів. Додати поточний вузол до списку опитаних.

Так як список вузлів, які передаються у відповіді, недетермінований (15 найближчих сусідів та 5 випадкових), то неможливо статично визначити кількість запитів, які потрібно надіслати, щоб отримати усіх сусідів поточного вузла. Тому для визначення кількості повідомлень використовується евристичний підхід: надсилати повідомлення доти, доки за останніх  $L$  повідомлень отримується новий вузол, але не менше  $M$  запитів.

Якщо ж вузол не надає відповіді на запит вузлів, це може означати, що встановлення з'єднання на кроці 4 не відбулось. У такому випадку повернутись до попереднього кроку;

Крок 6. Повернутись до кроку 3, якщо черга не порожня. Якщо порожня – повернути список усіх отриманих вузлів.

Також можливе вдосконалення на даному кроці. Так як блокчейн-мережа динамічна, то до неї постійно підключаються та відключаються вузли. Тому можливе опитування уже оброблених вузлів для повторного отримання вузлів з певною періодичністю.

### 2.2.3 Формалізоване представлення методу

Розглянемо блокчейн мережу Rootstock як орієнтований граф  $G = (V, E)$

$V$  – множина блокчейн-вузлів,

$E$  – множина напрямлених ребр, де кожне ребро  $(v_i, v_j)$  вказує на наявність зв'язку від вузла  $v_i$  до вузла  $v_j$ .

Кожен вузол  $v_i$  зберігає обмежений набір сусідніх вузлів,  $N_{v_i} \subset V$  але  $N_{v_i} \neq V$ . Використовуючи алгоритм пошуку у ширину, виконується обхід кожного блокчейн-вузла, починаючи з завантажувальних вузлів. Крім того, так як граф

динамічний (вузли постійно під'єднуються до мережі та від'єднуються), то для отримання оновленої інформації відвідані вузли потрібно повторно опитувати.

Процес описаний наступним чином:

1. Ініціалізація:  $Q = \{v_0\}, X = \{\}$ ,  $Q$  – черга вузлів для опитування,  $X$  – множина опитаних вузлів,  $v_0$  – завантажувальний вузол.
2. Доки  $Q$  не порожня,  $Q \neq \emptyset$ :
  - 2.1 Дістати вузол  $v_i$  з черги  $Q$ ,  $v_i \in Q$ ;
  - 2.2 Застосувати функцію  $R(v_i)$  для отримання сусідніх вузлів  $N_{v_i}$ ,  $R(v_i) \subseteq N_{v_i}$ ;
  - 2.3 Серед усіх  $R(v_i)$  додати до черги  $Q$  ті, які не помічені як опитані,  $R(v_i) \setminus X$ ;
  - 2.4 Помітити вузол  $v_i$  як опитаний,  $v_i \in X$ .
3. Випадково обрати опитаний вузол серед вже опитаних  $v \in X$  та додати його до черги  $Q$ ,  $Q = \{v\}$ .
4. Перейти до кроку 2 або вихід.

Один запит на надання сусідніх вузлів у протоколі пошуку вузлів мережі Rootstock надає підмножину  $n$  найближчих та  $r$  випадкових сусідніх вузлів  $L(N_{v_i}, n, r) \subseteq N_{v_i}$ . Тому застосовується функція  $R(v_i)$ , яка намагається отримати усю множину  $N_{v_i}$  та включає наступні дії:

1. Ініціалізація:  $T = \{\}$  – множина отриманих сусідів вузла  $v_i$ .
2. Здійснити підключення до  $v_i$ , функція  $C(v_i)$ :
  - 2.1 Надіслати PING-запит;
  - 2.2 Отримати PONG-відповідь;
  - 2.3 Отримати PING-запит;
  - 2.4 Надіслати PONG-відповідь.
3. Отримати сусідні вузли  $v_i$ , функція  $G(v_i, m, p)$ :
  - 3.1 Встановити лічильник кількості запитів  $k = 0$ ;
  - 3.2 Надіслати FIND\_NODE-запит та збільшити лічильник  $k = k + 1$ ;

3.3 Отримати відповідь NEIGHBORS, яка містить  $L(v_i, n, r)$  та додати її до  $T$ ,  $L(v_i, n, r) \subseteq T$ . Відсутність NEIGHBORS-відповіді означає, що з'єднання з вузлом  $v_i$  відсутнє (з боку вузла  $v_i$ ), тому якщо це був перший запит ( $k = 1$ ), то перейти до кроку 2;

3.4 Якщо кількість надісланих запитів  $k < m$ , перейти до кроку 3.2;

3.5 Якщо за останні  $p$  запитів отримано нові сусідні вузли, тобто  $|T_k| > |T_{k-p}|$ , то перейти до кроку 3.2.

4. Повернути  $T$ ,  $T \subseteq N_{v_i}$ .

Задля зменшення кількості спроб встановлення з'єднання з вузлом  $v_i$  розроблена також оптимізована функція  $R_{\text{опт}}$ . Вона відрізняється від  $R$  тим, як здійснюється підключення до вузла, а саме, функцією  $C$ . Функція  $C_{\text{опт}}(v_i, q)$  включає наступні кроки:

1. Надіслати PING-запит.
2. Отримати PONG-відповідь.
3. Застосувати функцію *esrecover* [155] та визначити публічний ключ (ідентифікатор)  $pub_{v_i}$  вузла  $v_i$ , на основі підпису отриманої PONG-відповіді.
4. Згенерувати випадковий приватний ключ  $priv$  та відповідний публічний ключ  $pub$ .
5. Якщо XOR-відстань з вузлом  $v_i$  більша за задане значення  $XOR(pub, pub_{v_i}) > q$ , повторити крок 4.
6. Встановити  $priv$  як свій приватний ключ та застосовувати його для подальшої комунікації
7. Надіслати PING-запит.
8. Отримати PONG-відповідь.
9. Отримати PING-запит.
10. Надіслати PONG-відповідь.

Модель:

$F: (v_0, R) \rightarrow V$  або  $F: (v_0, R_{\text{опт}}) \rightarrow V$

$R: (v_i, m, p) \rightarrow N_{v_i}$

$R_{\text{опт}}: (v_i, m, p, q) \rightarrow N_{v_i}$

де

$F$  – функція отримання вузлів блокчейн мережі Rootstock.

$v_0$  – початковий (завантажувальний) вузол, з якого починається пошук.

$R$  – функція отримання сусідніх вузлів з одного вузла.

$R_{\text{опт}}$  – оптимізована функція отримання сусідніх вузлів з одного вузла.

$v_i$  – поточний вузол, який опитується.

$m$  – мінімальна кількість запитів до одного вузла для отримання сусідніх.

$p$  – кількість останніх FIND\_NODE-запитів, які надали хоча б одного нового сусіда, щоб продовжити опитувати поточний вузол.

$q$  – максимальна XOR-відстань до вузла  $v_i$ .

$N_{v_i}$  – вихідна множина сусідів вузла  $v_i$ .

$V$  – вихідна множина знайдених вузлів мережі.

## **2.3 Метод визначення відправника транзакцій за допомогою псевдовузлів**

Метод полягає у розгортанні псевдовузла (або декількох), який під'єднується до кожного вузла мережі Rootstock та прослуховує усі трансльовані транзакції. Прослуховування транзакцій з кожного вузла усієї блокчейн мережі дозволяє виявити, хто трансльовав транзакцію першим. Під псевдовузлом розуміється вузол, який лише прослуховує транзакції та не виконує основної логіки.

### **2.3.1 Особливості Wire протоколу**

Реалізація Wire протоколу для вузла Rootstock мережі передбачає наступну конфігурацію [156]:

– максимальна кількість підключень до вузла 30 (*maxActivePeers*) за замовчуванням;

- максимальна кількість підключень з одного адресного блоку 16 (*filter.maxConnections*);
- безкласова маршрутизація блоку 16 (*filter.networkCidr*);
- асиметричне шифрування для обміну ключами та симетричне шифрування для повідомлень.

Враховуючи задані параметри, для реалізації методу потрібно врахувати наступне:

- при невдалій спробі підключення до вузла (отримання помилки *TOO\_MANY\_PEERS*) підключитись через деякий час;
- при невдалій спробі підключення до вузла (забагато підключень з одного блоку адрес) підключитись використовуючи іншу IP-адресу.

Першим етапом встановлення з'єднання вузлів по Wire протоколу є хендшейк, під час якого відбувається обмін ключами для встановлення шифрованого каналу.

На рис. 2.7 зображена діаграма об'єктів реалізації Wire протоколу у RSKj. На початку встановлення з'єднання послідовність обробників повідомлень складається з *ReadTimeoutHandler* та *HandshakeHandler*.

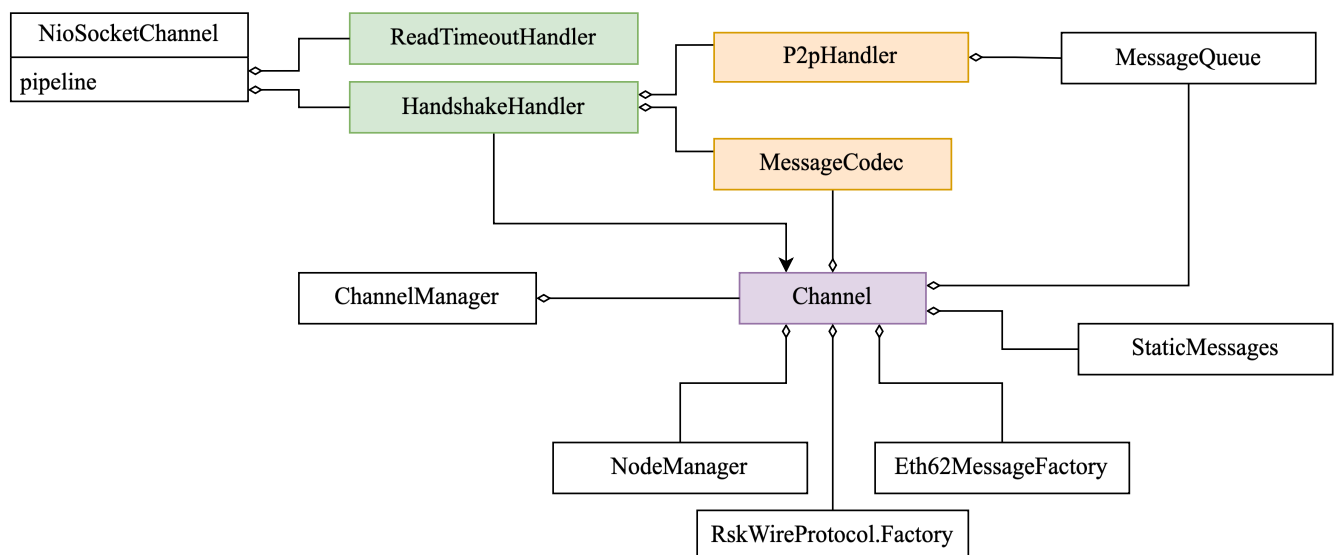


Рис. 2.7. Діаграма об'єктів реалізації Wire протоколу

Після встановлення хендшейку з пайплайну видаляється обробник HandshakeHandler та додаються наступні (рис. 2.8): FrameCodecHandler, MessageCodec, P2pHandler, RskWireProtocol.

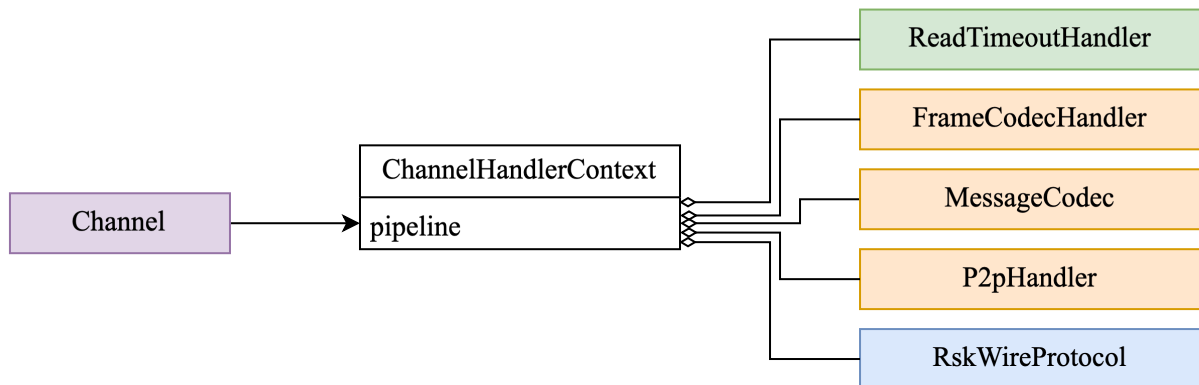


Рис. 2.8. Діаграма об'єктів реалізації Wire протоколу після хендшейку

RskWireProtocol обробляє 2 типи команд [157]:

- *STATUS*;
- *RSK\_MESSAGE*.

Команда *RSK\_MESSAGE* передбачає наступні типи повідомлень [158]:

- *STATUS\_MESSAGE*;
- *BLOCK\_MESSAGE*;
- *GET\_BLOCK\_MESSAGE*;
- *NEW\_BLOCK\_HASHES*;
- *TRANSACTIONS*;
- *BLOCK\_HASH\_REQUEST\_MESSAGE*;
- *BLOCK\_HASH\_RESPONSE\_MESSAGE*;
- *BLOCK\_HEADERS\_REQUEST\_MESSAGE*;
- *BLOCK\_HEADERS\_RESPONSE\_MESSAGE*;
- *BLOCK\_REQUEST\_MESSAGE*;
- *BLOCK\_RESPONSE\_MESSAGE*;

- *SKELETON\_RESPONSE\_MESSAGE*;
- *BODY\_REQUEST\_MESSAGE*;
- *BODY\_RESPONSE\_MESSAGE*;
- *SKELETON\_REQUEST\_MESSAGE*;
- *NEW\_BLOCK\_HASH\_MESSAGE*.

### 2.3.2 Основні етапи методу

Метод визначення відправника транзакцій за допомогою псевдовузлів передбачає застосування методу виявлення вузлів мережі, розглянутий у розділі 2.2. Метод включає наступні дії:

Крок 1. Застосування методу виявлення усіх доступних вузлів мережі. Даний етап виконується на рівні протоколу Peer Discovery. Вказуються завантажувальні вузли, з яких починається пошук інших вузлів (рис. 2.9 та рис. 2.10). Кожен вузол надає сусідні вузли для псевдовузла, який їх запитує. Після підключення до сусіднього вузла (рис. 2.11), сусідній вузол надає інші сусідні вузли (рис. 2.12) і так далі, доки не буде виявлено усі вузли мережі. Кожен вузол вважає, що він здійснює підключення до звичайного вузла, хоча насправді це псевдовузол та не реалізує основної логіки.

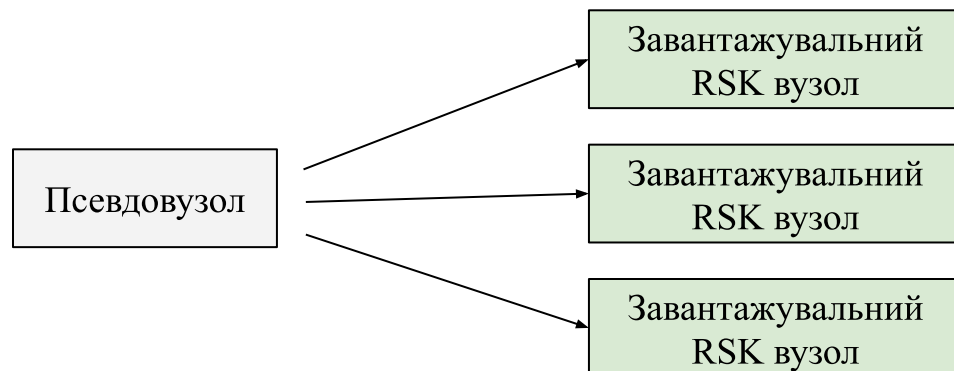


Рис. 2.9. Задання початкових вузлів



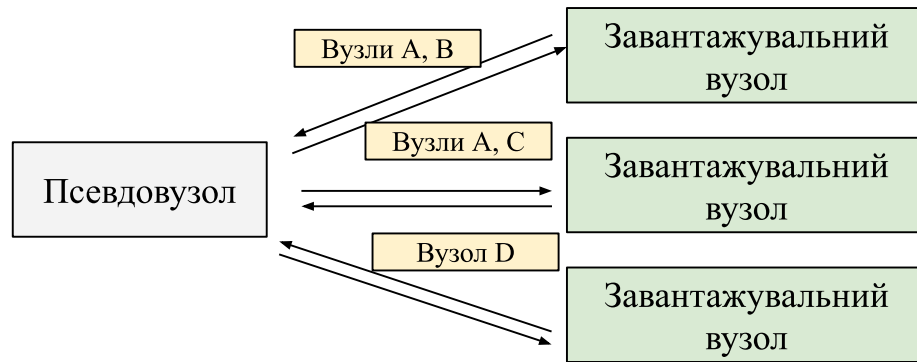


Рис. 2.10. Завантажувальні вузли надають список інших вузлів

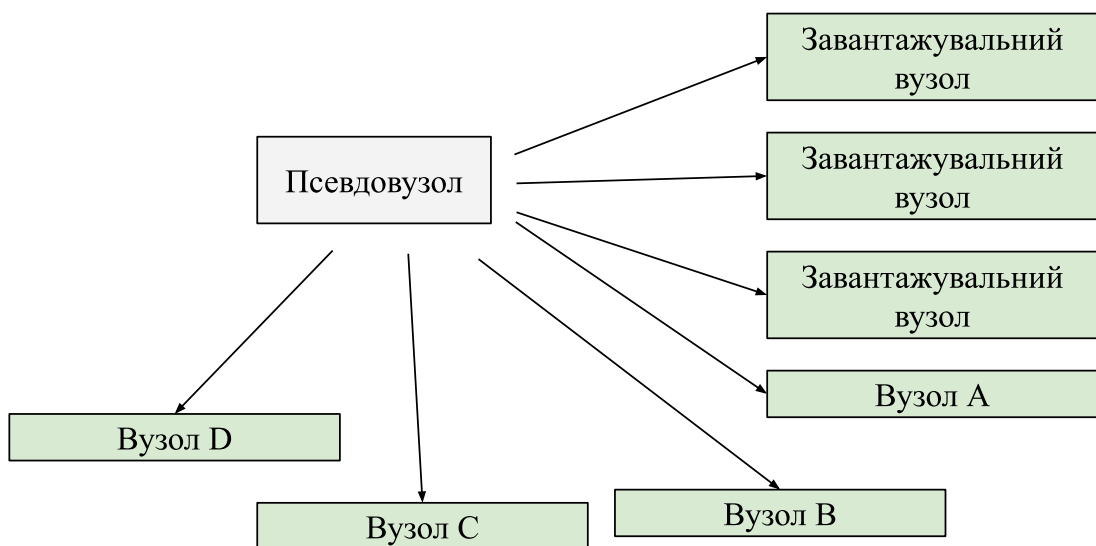


Рис. 2.11. Підключення до виявлених вузлів

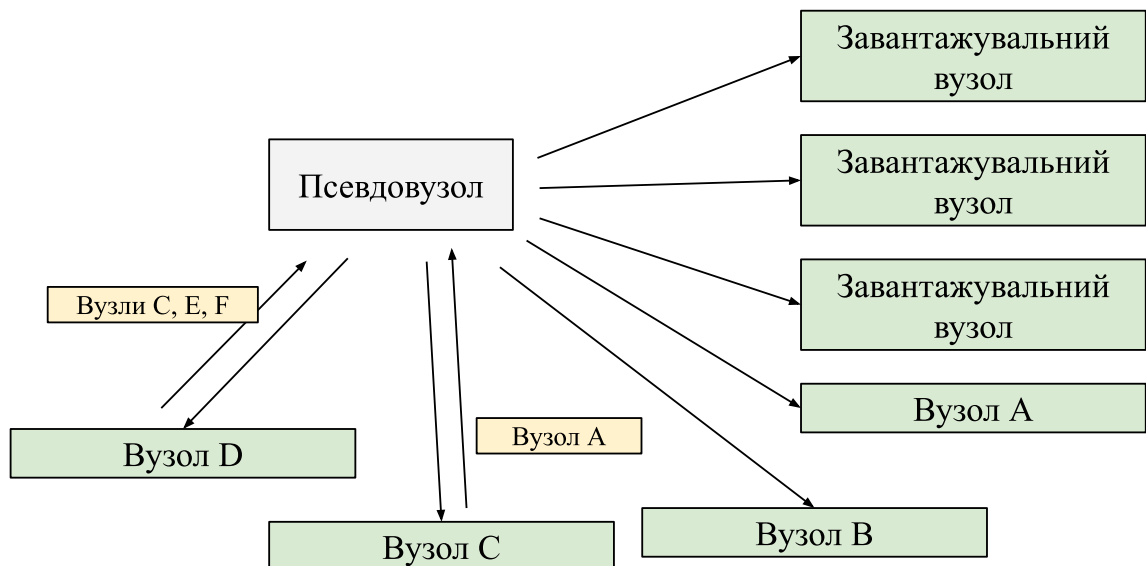


Рис. 2.12. Вузли надають сусідні вузли

Крок 2. Здійснюється підключення до кожного вузла по Wire протоколу та прослуховування усіх повідомлень типу *TRANSACTIONS*. При трансляції транзакції з вузла мережі до псевдовузла (рис. 2.13) зберігається час, коли і яким вузлом була надіслана транзакція.

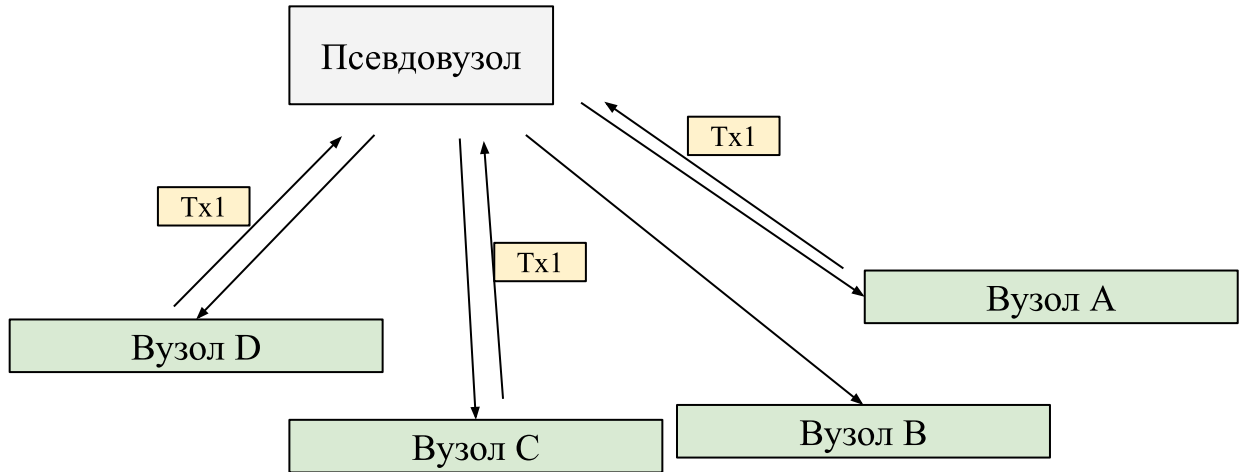


Рис. 2.13. Вузли надсилають транзакції до псевдовузла (системи виявлення відправника транзакції)

Крок 3. Визначення вузла, який перший транслював запитану транзакцію. Серед усіх записів про трансляцію транзакцій, обирається вузол, який першим її надіслав. Надається інформація про IP-адресу вузла, публічний ключ (ідентифікатор) вузла, час отримання транзакції та її відправник (той, хто підписав).

### 2.3.3 Формалізоване представлення методу

Процес визначення вузла, з якого була відправлена транзакція, описаний наступним чином:

1. Для кожного блокчейн-вузла  $v_i \in V$  здійснити підключення використовуючи порт  $p_i \in P$ :
  - 1.1 Ініціювати рукоштовування (handshake) з вузлом;
  - 1.2 Встановити шифрований канал;
  - 1.3 Почати процес отримання повідомлень від вузла.

2. Для кожної отриманої транзакції  $x \in X$  зберегти час її транслювання вузлом  $v$ ,  $t_{xv} \in T_x$ .

3. Для визначення відправника транзакції  $x$  знайдемо  $v^*$  таке, що  $t_{xv^*}$  є мінімальним серед усіх  $t_{xv_i} \in T_x$ . Тобто,  $v^* = v_i$ , для якого  $t_{xv_i} = \min(T_x)$ .

Модель:

$F_1: (V, P) \rightarrow T$

$F_2: (T, x) \rightarrow v$

де

$F_1$  – функція підключення з кожним вузлом мережі та встановлення часу надсилання транзакцій кожним вузлом.

$F_2$  – функція визначення вузла, що першим відправив транзакцію.

$V$  – множина вузлів блокчейн-мережі Rootstock (отримано за допомогою методу 2).

$T$  – вихідна множина часу надсилання кожної транзакції кожним вузлом.

$x$  – транзакція, відправника якої потрібно визначити.

$v$  – вихідний вузол, який першим транслював транзакцію  $x$ .

## 2.4 Метод аналізу OP\_RETURN-скриптів Bitcoin мережі

Метод полягає у створенні бази даних форматів OP\_RETURN (*nulldata*) скриптів на основі частоти появи їх префіксних частин.

### 2.4.1 Особливості OP\_RETURN-скриптів Bitcoin мережі

У Bitcoin скриптах OP\_RETURN опкод слугує механізмом для позначення виходу транзакції як та, що не може бути витрачена. Це робить даний вихід непридатним для використання його як входу у наступних транзакціях. Основне призначення цього опкоду полягає у додаванні даних у блокчейн як частину транзакції Bitcoin. Використання OP\_RETURN для даних має широкий спектр застосувань, включаючи перевірку існування документа, вставку метаданих для

різних протоколів (надбудов) та для зберігання даних у мережі Bitcoin. Опкод *OP\_RETURN* дозволяє використовувати 80 байтів довільних даних.

У роботі [124] досліджується мережа Bitcoin з метою аналізу та класифікації стандартних і нестандартних транзакцій. У роботі також робиться огляд форматів та довжин *OP\_RETURN*-скриптів:

- *OP\_RETURN* <hash> скрипт, де *hash* це будь-який хеш будь-яких даних;
- *OP\_RETURN* <error> скрипт, де *error* це кількість байтів, які потрібно покласти до стеку. В даному випадку кількість байтів не відповідає довжині скрипта, тому виникає помилка і скрипт вважається нестандартним. У статті зазначається, що ця транзакція відбулася скоріше за все через програмну помилку;

У роботі [122] здійснений вичерпний аналіз форматів та префіксів *OP\_RETURN*-скриптів, а також даних, які зберігаються у даних скриптах.

#### **2.4.2 Основні етапи методу**

Метод аналізу *OP\_RETURN*-скриптів передбачає аналіз вихідних скриптів та включає наступні дії.

Крок 1. Синхронізація Bitcoin-вузла та виокремлення транзакцій з *OP\_RETURN*-виходами. Збереження усіх *OP\_RETURN* скриптів у дизасембльованому представленні.

Крок 2. Збір усіх можливих префіксів з дизасембльованих скриптів (враховуючи задану мінімальну та максимальну довжиною) і частоти їх появи.

Крок 3. Виокремлення префіксів на основі усіх зібраних. Серед усіх префіксів обираються ті, які зустрічаються хоча б задану мінімальну кількість разів. На даному етапі важливо виокремити спільні префікси, намагаючись покрити максимальну кількість існуючих скриптів одним префіксом. В той же час важливо зберегти варіативність типів (розпізнаних форматів). Розроблений алгоритм полягає у сортуванні усіх скриптів у алфавітному порядку і аналізі відношення кількості появи поточного префіксу до наступного. Виокремлені префікси зберігаються.

Крок 4. Застосування виокремлених префіксів до нових OP\_RETURN транзакцій для визначення їх типів.

#### 2.4.5 Формалізоване представлення методу

Процес визначення префіксів скриптів включає наступні дії:

1. Попередня обробка OP\_RETURN-скриптів, представлених у формі опкодів. Для кожного скрипта  $s \in S$ :

1.1 Видалити частину “OP\_RETURN”;

1.2 Видалити символи “ ” (пробіл, який розділяє дані, що додаються до стеку).

1.3 Застосувати функцію  $U(s)$  для конвертації ASCII-представлення скрипта у представлення байтів.

2. Збір можливих префіксів з заданою мінімальною та максимальною довжиною  $m$  та  $n$  відповідно:

2.1 Визначити функцію  $E(s, i)$ , що виділяє послідовність символів  $e$  довжиною  $i$  з послідовності  $s$ , тобто  $e \subseteq s$ ;

2.2. Створити список префіксів  $G = \{\}$ , елементами якого будуть пари префіксу та кількості його використання;

2.3. Для кожного скрипта  $s \in S$  та для усіх можливих довжин  $l$ , такі що  $m < l < \min(n, |s|)$ , застосувати функцію  $e = E(s, l)$ , де  $|s|$  – довжина скрипта. Якщо префікс  $e$  міститься у списку  $(e, cnt_e) \in G$ , то збільшити кількість використання  $cnt_e$  на 1. Інакше додати елемент  $(e, 1)$  до  $G$ ;

3. Видалити елементи зі списку  $(prf, cnt) \in G$ , для яких  $cnt < k$ , де  $k$  – мінімальна кількість використання префіксу.

4. Відсортувати список  $G$  по префіксу, тобто  $G = \{(prf_0, cnt_0), (prf_1, cnt_1), \dots\}$ , де  $prf_i < prf_{i+1}$ .

5. Для виокремлення префіксів серед префіксів усіх можливих довжин застосувати функцію  $R(G, q)$ .

Функція  $R(G, q)$  обробляє префікси можливих довжин та виділяє ті, довжина яких не перевищує наступний префікс у  $q$  разів:

1. Створити список виокремлених префіксів  $P = \{(prf_0, cnt_0)\}$ , де  $(prf_0, cnt_0)$  – перший елемент списку  $G$ .
2. Для кожного елемента  $(prf, cnt) \in G$ :
  - 2.1 Дістати та видалити зі списку  $P$  останній доданий елемент  $(prf_{prev}, cnt_{prev})$
  - 2.2 Якщо  $prf_{prev}$  не є префіксом елемента  $prf$ , то додати елементи  $(prf_{prev}, cnt_{prev})$  та  $(prf, cnt)$  до списку  $P$ ;
  - 2.3 Інакше, якщо  $\frac{cnt}{cnt_{prev}} < q$ , додати до списку  $P$  тільки  $(prf_{prev}, cnt_{prev})$ ;
  - 2.4 Інакше додати до списку  $P$  тільки  $(prf, cnt)$ ;
3. Повернути список  $P$ ,  $P \subset G$ .

Модель:

$$F_1: (S, q, m, n, k) \rightarrow P$$

$$F_2: (P, s) \rightarrow p$$

$F_1$  – функція побудови множини префіксів.

$F_2$  – функція визначення типу OP\_RETURN-скрипта.

$S$  – множина усіх відомих OP\_RETURN-скриптів.

$q$  – коефіцієнт, який задає у скільки разів використання поточного префіксу повинно бути меншим використання попереднього префіксу для того, щоб попередній префікс вважався спільним.

$m$  – мінімальна довжина префіксу.

$n$  – максимальна довжина префіксу.

$k$  – мінімальна кількість використання префіксу.

$P$  – множина виокремлених префіксів.

$s$  – OP\_RETURN-скрипт, префікс (тип) якого потрібно визначити.

$p$  – розпізнаний тип скрипта.

## 2.5 Висновки до розділу 2

Розділ містить розроблене алгоритмічне забезпечення запропонованих методів аналізу блокчейн транзакцій.

Визначено основні етапи методу застосування блокчейн-провідників для отримання даних з різних блокчейн-мереж для подальшого блокчейн аналізу, а саме:

- вибір блокчейн-провідника;
- аналіз архітектури існуючої або нової системи аналізу блокчейн-мереж, в яку буде інтегрована програмна реалізація методу;
- визначення способу комунікації з блокчейн-провідником;
- програмна реалізація отримання даних з блокчейн-провідників.

Проаналізовано існуючі блокчейн-провідники та обрано Blockscout як оптимальний для використання у даному дослідженні. Даний провідник доступний для більше ніж 200 блокчейн-мереж, має відкритий вихідний код та може надавати дані про розгорнутий EVM смарт-контракт, включаючи його байт-код, вихідний код (якщо наявний) та деталі проксі-контракту.

Проаналізовано Peer Discovery протокол блокчейн мережі Rootstock. Використано комплексний підхід, що включає в себе аналіз вихідного коду програмного забезпечення вузла RSKj та аналітичний огляд документації концептуально схожого протоколу – протоколу виявлення вузлів в мережі Ethereum. Це дозволило зрозуміти як реалізована та працює розподілена таблиця вузлів Kademlia, отримати типи і формати повідомлень та визначити послідовності повідомлень для отримання списку сусідніх вузлів. Іншими словами, відновлено повну логіку протоколу пошуку вузлів. Даної інформації достатньо для незалежної реалізації протоколу та комунікації у блокчейн мережі Rootstock з іншими вузлами.

Визначено основні етапи методу виявлення вузлів блокчейн мережі Rootstock на основі пошуку у ширину. Вузли опитуються по Peer Discovery протоколу для отримання сусідніх вузлів. Потім сусідні вузли опитуються для отримання їх сусідніх вузлів. Дана процедура продовжується, доки виявляються нові вузли. Також пошук може виконуватись у постійному режимі (вузли повторно

опитуються) для отримання інформації про стан мережі в реальному часі. Окрім базового алгоритму розроблена оптимізація потрапляння вузла у слот меншої відстані розподіленої таблиці шляхом підбору публічного ключа (ідентифікатора) вузла. Це дозволяє пришвидшити пошук вузлів за рахунок збільшення шансів того, що визначений слот таблиці буде незаповненим, і підключення відбудеться з першої спроби.

Проаналізовано Wire протокол блокчейн мережі Rootstock. Використано комплексний підхід, що включає в себе аналіз вихідного коду програмного забезпечення вузла RSKj. Відновлена логіка протоколу етапу встановлення з'єднання (хендшейку) та транслявання чи отримання повідомлень про транзакції. Даної інформації достатньо для незалежної реалізації частини Wire протоколу для пошуку оригінального відправника транзакції.

Визначено основні етапи методу визначення відправника транзакцій блокчейн мережі Rootstock. Даний метод передбачає застосування попереднього методу виявлення вузлів та наступне підключення до кожного по Wire протоколу. Після підключення прослуховуються транзакції від усіх вузлів. Для визначення вузла-відправника вказаної транзакції визначається вузол, який раніше інших вузлів здійснив трансляцію транзакції.

Проаналізовано особливості OP\_RETURN-скриптів, їхні формати та їх застосування у блокчейн мережі Bitcoin. На основі існуючої інформації скомпільована база форматів OP\_RETURN-скриптів, які використовувались у Bitcoin мережі. Описане застосування розробленого підходу для виокремлення форматів скриптів та створення відповідної бази форматів. Визначені основні етапи методу аналізу OP\_RETURN-скриптів з використанням отриманих баз форматів.



## РОЗДІЛ 3 ЗАСОБИ РОЗРОБКИ ТА ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ

### 3.1 Архітектура програмної системи аналізу блокчейн транзакцій

Запропонована архітектура системи аналізу блокчейн транзакцій зображена на рис. 3.1. Архітектура системи підтримує різні типи блокчейн-мереж.

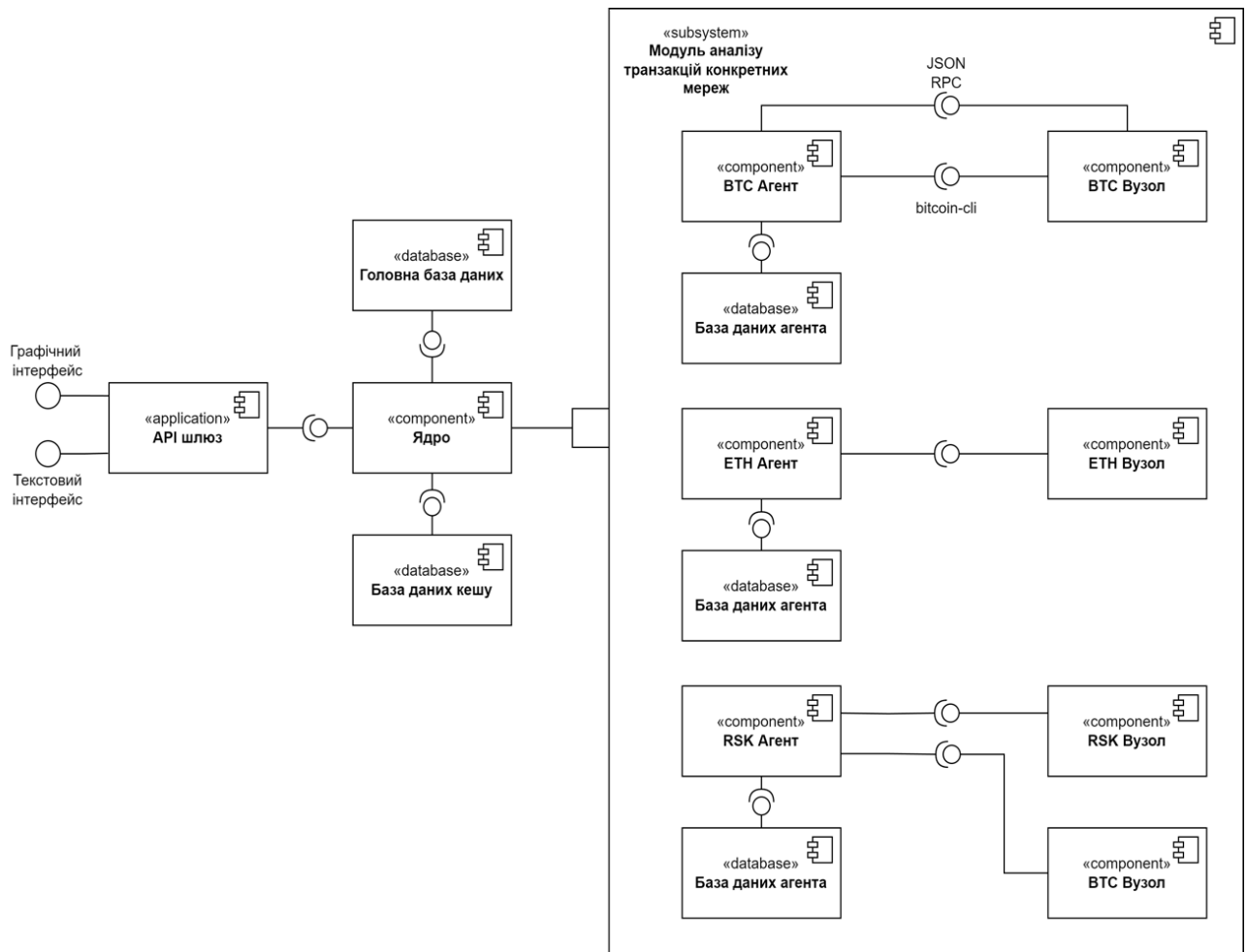


Рис. 3.1. Архітектура системи аналізу блокчейн транзакцій

Компоненти системи включають:

- Графічний та консольний інтерфейси – для роботи з системою;
- API шлюз – веб-інтерфейс прикладного програмування, який дозволяє комунікувати з ядром системи;
- Ядро системи – керує усією взаємодією: обробляє запити користувача, а також отримує та обробляє інформацію від агентів;

- Головна база даних – зберігає основну інформацію, параметри, а також дані про клієнтів та агентів;

- База даних кешу – зберігає тимчасову інформацію;

- Агент – елемент, який відповідає за управління вузлом конкретної мережі.

Для сайдчейн мереж (таких як RSK) агент управляє декількома вузлами. Агент реалізує обов’язковий функціонал (такий як отримання інформації про адресу, пошук транзакції), а також додатковий, який є специфічним для конкретної мережі;

- База даних агента – допоміжна база даних, яку використовує агент для зберігання обробленої інформації з конкретної мережі, а також різні налаштування;

- Вузол блокчейн мережі – розгорнутий вузол.

Взаємодія між ядром та агентами відбувається за принципом інверсія керування (Inversion of Control – IoC). Агенти, в тому числі розгорнуті на різних машинах, здійснюють підключення до ядра та надають ядру інтерфейс для взаємодії з блокчейн-мережею.

Розроблені методи можуть бути інтегровані у дану архітектуру. Інтеграція зображена на рис. 3.2.

Розроблені методи передбачають додаткові компоненти:

- Агент провідників – агент для застосування блокчейн-провідників для збору даних з різних блокчейн мереж. Агент комунікує з провідниками різних блокчейн-мереж;

- RSK Агент псевдовузлів – агент для розгортання та управління псевдовузлами (один або більше), які використовуються для визначення відправника транзакцій у Rootstock мережі;

- BTC Агент, База скриптів – модифікований агент для Bitcoin мережі та база даних OP\_RETURN-скриптів (форматів, структур та описів).

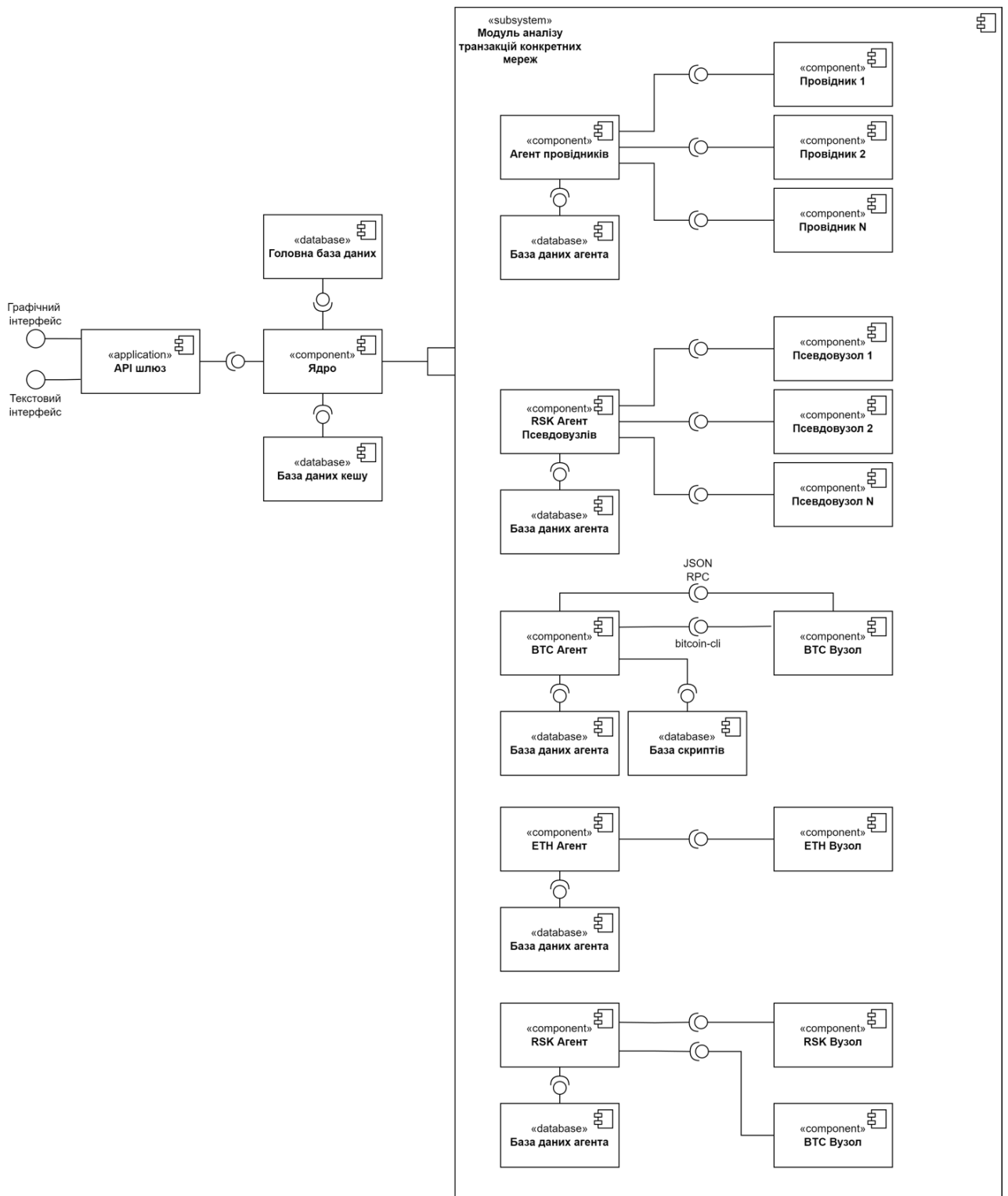


Рис. 3.2. Архітектура системи аналізу блокчейн транзакцій з урахуванням розроблених методів

### 3.2 Засоби розробки

Архітектура системи дозволяє використовувати мікросервісний підхід та реалізовувати компоненти, використовуючи різні операційні системи, мови

програмування, бібліотеки та бази даних. Блокчейн-вузли натомість можуть мати строгі вимоги до операційної системи та машини, на якій розгортається вузол.

У даній роботі для програмної реалізації розроблених методів використовується мова програмування Python.

### **3.3 Програмне забезпечення застосування блокчейн-провідників для збору даних з різних мереж**

Прототип методу побудований на основі структури розробленої у попередньому розділі. Структура передбачає одночасне отримання даних з декількох блокчейн-мереж для аналізу смарт-контрактів.

Для прототипу використано мову програмування Python, а також бібліотеку Beautiful Soup [159] для парсингу мови розмітки гіпертексту (HyperText Markup Language – HTML). Ці засоби є ефективними для прототипування та перевірки гіпотез.

В рамках цього процесу збору даних, очевидно, що можуть бути ситуації, коли вихідний код контрактів може бути недоступний. У таких випадках використано метод, відомий як декомпіляція байт-коду, для отримання псевдокоду.

Декомпіляція байт-коду – це процес зворотного інжинірингу виконавчого байт-коду, який представляє низькорівневі інструкції смарт-контракту. Застосовуючи спеціалізовані інструменти та алгоритми, байт-код перетворюється назад у більш зрозумілу для людини форму, подібну до структури та логіки початкового вихідного коду. Це дозволяє отримати деталі щодо основного функціоналу, деталей реалізації та можливих вразливостей смарт-контракту, незважаючи на відсутність початкового вихідного коду.

Наразі був обраний декомпілятор Panoramix decompiler [160], оскільки він активно підтримується, має відкритий вихідний код, є стабільним і дає прийнятні результати. Panoramix перетворює байт-код EVM на псевдокод мови Python.

Для того, щоб отримати дані з Blockscout, потрібно визначити, які посилання відповідають за надання певних даних. В рамках цього дослідження використовуються лише три ендпоїнти Blockscout провідника:

– /accounts – для отримання адреси облікового запису, балансу криптовалюти і типів облікових записів, будь то смарт-контракт або зовнішній обліковий запис (Externally Owned Account – EOA). Даний ендпоїнт надає лише топ-акаунти (облікові записи), баланс криптовалюти яких перевищує 0. Варто зазначити, що native-контракти (native smart contracts – інтегровані у блокчейн систему, а не розгорнуті у ній) можуть бути розпізнані як EOA, а не як смарт-контракти;

– /address/{address}/contracts – для отримання вихідного коду смарт-контракту (якщо він доступний) та розгорнутого байт-коду. Крім того, якщо контракт є контрактом-проксі [161], Blockscout виявляє його і надає адресу реалізації;

– /address/{address}/token-balances – для отримання токенів, які належать обліковому запису, та їх кількість.

Функції парсингу зображені на рисунках 3.3-3.5.

```
def parse_account(html):
    isContract = False
    soup = BeautifulSoup(html, 'html.parser')
    item = soup.find('span', {'class': ''})
    addr = item.get('data-address-hash')
    if not addr:
        isContract = True
        item = soup.find('span', {'class': 'contract-address'})
        addr = item['data-address-hash']

    value = item.find_next('span', {'data-test': 'address_balance'}).string
    value = value.strip().replace(',', '')
    value = float(value.split())[0]

    return (addr, value, isContract)
```

Рис. 3.3. Функція парсингу сторінки облікових записів

```

soup = BeautifulSoup(r.content, 'html.parser')
item = soup.find('dd', {'data-test': 'address_contract_implementation'})
if item:
    item = item.find('a', {'class': 'do_fetch_contractscontract-address'})
    implementation = item['href'].split('/')[1].strip()
else:
    implementation = ''
implementations.append(implementation)

buttons = soup.find_all('button', {'id': 'button'})
i = 0
for button in buttons:
    ext = {'Copy Source Code': 'sol', 'Copy Deployed ByteCode': 'bin'}.get(button.string.strip())
    if not ext:
        continue
    contract = button['data-clipboard-text']
    if ext == 'bin':
        contract = contract[2:]
        name = 'deployed'
        total_size += len(contract) // 2
        contracts_count += 1
    else:
        name = i
        i += 1

```

Рис. 3.4. Функція парсингу сторінки смарт-контракту

```

soup = BeautifulSoup(r.content, 'html.parser')
for item in soup.find_all('p', {'class': 'dropdown-amount'}):
    item = item.string.replace(',', '').strip()
    amount, token = item.split()
    amount = float(amount)
    tokens.append((token, amount))

```

Рис. 3.5. Функція парсингу сторінки балансів токенів

Отримання даних, парсинг і обробка можуть бути паралелізовані за допомогою пулу потоків або пулу процесів [162]. Спрощений процес паралельного отримання даних показано на рис. 3.6.

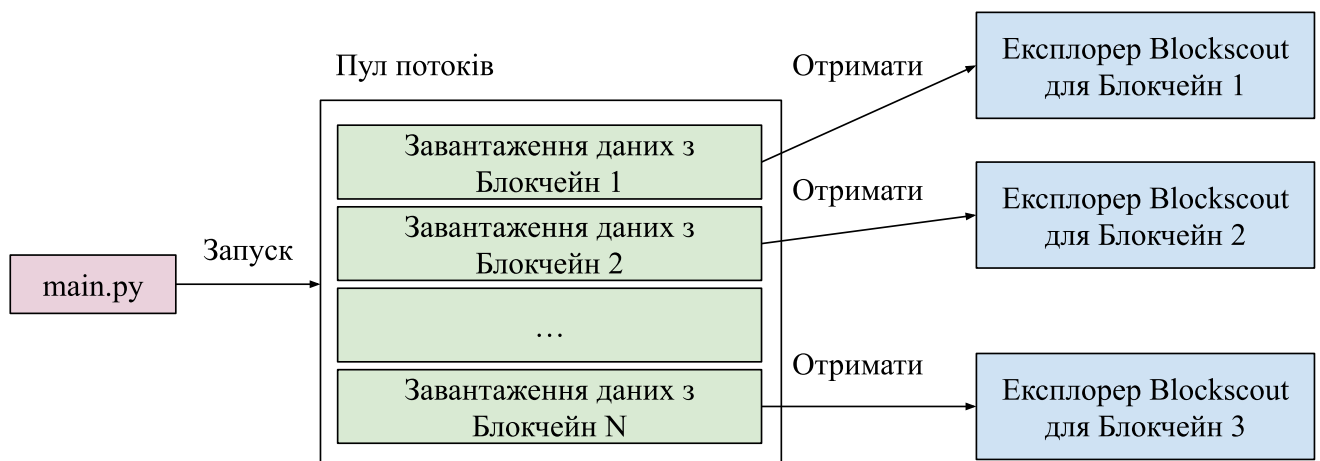


Рис. 3.6. Паралельне отримання даних за допомогою пулу потоків

Це дозволить уникнути різних вузьких місць, серед яких основними є затримка мережі та декомпіляція байт-коду смарт-контракту.

### 3.4 Програмне забезпечення виявлення та ідентифікації блокчейн вузлів

Прототип програмного забезпечення виявлення вузлів реалізований мовою Python. Для пришвидшення пошуку алгоритм реалізований багатопоточно з використанням засобів ThreadPoolExecutor та queue (синхронізована черга). Для асиметричної криптографії, а саме для генерації приватного ключа, підпису повідомлення та інших дій використовується бібліотека coincurve. Для хешування використовується бібліотека ruscryptodome.

Програмне забезпечення здійснює підключення до декількох вузлів одночасно у 15 різних потоках та запитує сусідні вузли. Для кращої ізоляції для різних вузлів використовується окремі порти (від 1000 до 9999). На рис. 3.7 зображена умовна схема, де програмне забезпечення підключається до декількох вузлів одночасно та використовує різні порти для отримання повідомлень.

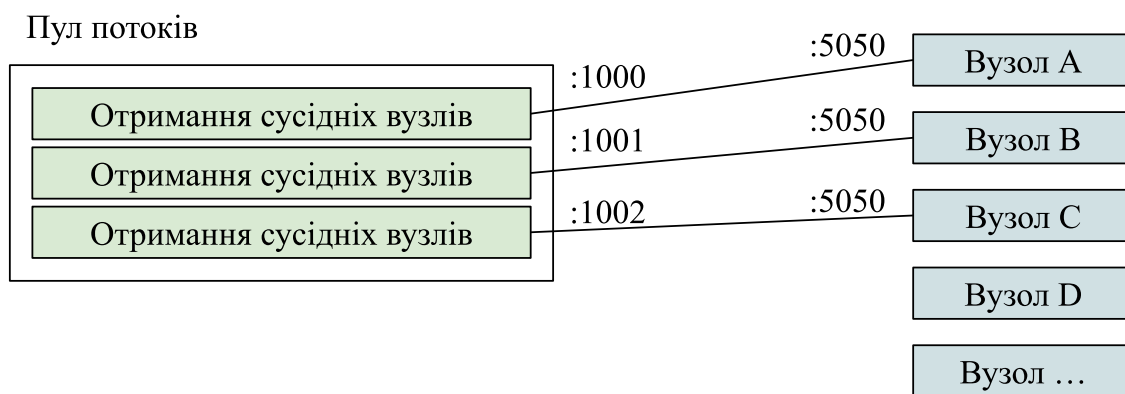


Рис. 3.7. Багатопотокова реалізація підключень

Програмне забезпечення методу передбачає надсилання запитів PING, FIND\_NODE, а також надсилання відповіді PONG. Відповідь NEIGHBORS не надсилається, а лише отримується та обробляється. У повідомленнях поле rlpCheck генерується як випадковий ідентифікатор UUID. Так як експеримент проводиться у основній мережі (mainnet), то ідентифікатор мережі rlpNetworkId задається 775.

При зборі вузлів реєструється наступна інформація:

- IP адреса вузла;
- UDP та TCP порти для Peer Discovery та Wire протоколів;
- ідентифікатор вузла;
- час виявлення вузла.

Для отримання геолокації IP-адреси використовується API веб-сервісу Geolocation DB.

### **3.5 Програмне забезпечення визначення відправника транзакцій за допомогою псевдовузлів**

Програмне забезпечення визначення відправника транзакцій передбачає розробку наступних компонентів:

1. Реалізація методу пошуку усіх доступних вузлів мережі RSK (представлено у розділі 3.4). Пошук вузлів здійснюється постійно.

2. Реалізація Wire протоколу:

- 2.1 Рукостискання, під час якого за допомогою криптографії на еліптичних кривих відбувається генерація спільного ключа, який буде використаний при симетричному шифруванні;

- 2.2 Реалізація комунікації між вузлами використовуючи шифрування AES в режимі CTR. З'єднання з вузлом повинно підтримуватись постійно;

- 2.3 Реалізація парсингу Wire-повідомлення типу *TRANSACTIONS*.

3. Реалізація функції збереження транзакцій та інформації про відправлення.

Для дано програмного забезпечення використовуються наступні Python бібліотеки: `rlp`, `coincurve`, `ruscryptodome`.

Проведення експеримента також передбачає розробку наступних скриптів:

- скрипт для розгортання перевірочних вузлів;
- скрипт для періодичного відправлення транзакцій до перевірочних вузлів.



### 3.6 Програмне забезпечення аналізу OP\_RETURN-скриптів Bitcoin мережі

В якості блокчейн вузла застосовується програмне забезпечення Bitcoin Core 26.0. Блокчейн-дані зберігаються на зовнішньому твердотілому накопичувачі. Для реалізації даного методу була обрана нереляційна база даних MongoDB. Робота з базою даних реалізується через бібліотеку `pymongo`. Візуалізація статистичних даних відбувається за допомогою бібліотеки `matplotlib`.

Після повної синхронізації вузла для доступу до блокчейн даних використовується його RPC інтерфейс.

Починаючи з блоку 0 кожен блок запитується за допомогою виклику `getblock`, передаючи хеш блоку та параметр `verbosity`, встановлений у 2. Дане значення параметру `verbosity` позначає включення даних транзакцій у відповідь, в тому числі їх шістнадцяткове та дизасембльоване представлення. Приклад отриманих даних представлений на рисунку 3.8.

Якщо транзакція містить тип вихідного скрипта `nulldata` (OP\_RETURN), то вона зберігається до бази даних. Наступним кроком є отримання вихідних OP\_RETURN скриптів зі збережених транзакцій та застосування алгоритму пошуку префіксів. На рис. 3.9 зображено реалізацію функції виокремлення префіксів на основі частоти їх появи. На вхід подаються, відсортовані в алфавітному порядку, усі можливі префікси, але обмежені мінімальною і максимальною довжиною та мінімальною кількістю появи.

```
"result": {
  "hash": "000000008bf44a528a09d203203a6a97c165cf53a92ecc27aed0b49b86a19564",
  "confirmations": 840371,
  "height": 1337,
  "version": 1,
  "versionHex": "00000001",
  "merkleroot": "bda142f78afd9ee444a9aaa7fd2a4e35f3e7d785d4e9498943bed752198d44b4",
  "time": 1232629728,
  "mediantime": 1232625123,
  "nonce": 3573931807,
  "bits": "1d00ffff",
  "difficulty": 1,
  "chainwork": "000000000000000000000000000000000000000000000000000000000000000053a053a053a",
  "nTx": 1,
  "previousblockhash": "0000000073b4019fc436e5a38368f24418eb773fb7d142644a47238e2ac423ce",
  "nextblockhash": "000000006bfee7e986540759c8b18085b8f6cdbbae8bdefe8e5bc12086c25777",
  "strippedsize": 216,
  "size": 216,
  "weight": 864,
  "tx": [
    {
      "txid": "bda142f78afd9ee444a9aaa7fd2a4e35f3e7d785d4e9498943bed752198d44b4",
      "hash": "bda142f78afd9ee444a9aaa7fd2a4e35f3e7d785d4e9498943bed752198d44b4",
      "version": 1,
      "size": 135,
      "vsize": 135,
      "weight": 540,
      "locktime": 0,
      "vin": [
        {
          "coinbase": "04ffff001d02ee00",
          "sequence": 4294967295
        }
      ],
      "vout": [
        {
          "value": 50,
          "n": 0,
          "scriptPubKey": {
            "asm":
045793aa1d737d8be6b6fbce8d78df2cf115d4424f38ec07c95ef7766be82a1a3f546db2cd13fd6dab2e077b8ec35af4e2c718724b
919e550927555758870be235 OP_CHECKSIG",
            "desc":
pk(045793aa1d737d8be6b6fbce8d78df2cf115d4424f38ec07c95ef7766be82a1a3f546db2cd13fd6dab2e077b8ec35af4e2c7187
24b919e550927555758870be235)#ehy2vf3l",
            "hex":
41045793aa1d737d8be6b6fbce8d78df2cf115d4424f38ec07c95ef7766be82a1a3f546db2cd13fd6dab2e077b8ec35af4e2c718724
```

```
def reduce_prefixes(prefixes):
    reduced = []
    reduced.append(prefixes[0])
    for value, count in prefixes:
        prev_value, prev_count = reduced.pop()
        if not value.startswith(prev_value):
            reduced.append((prev_value, prev_count))
            reduced.append((value, count))
        elif count / prev_count < q:
            reduced.append((prev_value, prev_count))
        else:
            reduced.append((value, count))
    return reduced
```

Рис. 3.9. Реалізація функції виокремлення префіксів на основі частоти їх появи

Після побудови бази даних префіксів, вона застосовується при аналізі нових або існуючих транзакцій.

### 3.7 Висновки до розділу 3

У розділі наведена архітектура комплексної системи аналізу блокчейн транзакцій, що здатна підтримувати різні типи блокчейн мереж та модульно додавати додаткові компоненти аналізу. Реалізації запропонованих методів можуть бути додані до такої системи.

Запропонована архітектура системи для отримання даних з блокчейн-провідника про розгорнуті смарт-контракти в блокчейн-мережі та для подальшого їх аналізу. Система включає серію блоків для отримання даних, їх обробку та збереження. Архітектура передбачає використання ChatGPT як асистента для аналізу смарт контрактів.

На основі заданої архітектури розроблений багатопотоковий прототип системи, що дозволяє отримувати дані одночасно з різних блокчейн-мереж, використовуючи публічно доступні екземпляри блокчейн-провідників. Система збирає дані про смарт-контракти для їх подальшого аналізу. Байт-коди розгорнутих смарт-контрактів декомпілюються в процесі.

Розділ описує також розробку прототипу програмного забезпечення для виявлення вузлів мережі Rootstock, який реалізує алгоритм пошуку вузлів з використанням багатопотокових механізмів. Це дозволяє здійснити

пришвидшений пошук вузлів. При зборі вузлів отримується IP-адреса вузла, його ідентифікатор (публічний ключ), порти протоколів Peer Discovery і Wire, та час виявлення вузла.

Також у розділі описане програмне забезпечення визначення відправника транзакцій та програмне забезпечення аналізу OP\_RETURN-скриптів Bitcoin мережі.

## **РОЗДІЛ 4 ЕКСПЕРИМЕНТАЛЬНИЙ АНАЛІЗ РОЗРОБЛЕНИХ МЕТОДІВ АНАЛІЗУ БЛОКЧЕЙН ТРАНЗАКЦІЙ**

### **4.1 Оцінка ефективності застосування блокчейн-провідників для збору даних з різних мереж**

#### **4.1.1 Параметри експерименту**

Обрані 14 блокчейн-мереж на основі віртуальної машини Ethereum, які будуть використані для експерименту зі збору інформації: Acala [163], Aves [164], Energy Web [165], Era Swap [166], Karura [167], Kava [168], MCH-verse [169], Nahmii [170], Neatiii [171], OASYS [172], Puppynet [173], Rootstock [174], SmartBCH [175], Xiden [176].

Для дослідження було використано віртуальну обчислювальну машину з платформи Vultr [177] з найменш можливою конфігурацією для обраної операційної системи. Це машина звичайної продуктивності з такими параметрами:

- процесор: 1 віртуальний процесор (virtual CPU) Intel Core;
- оперативна пам'ять: 1 ГБ;
- твердотілий накопичувач: 25 ГБ;
- об'єм вихідного трафіку: 1 ТБ;
- операційна система: Ubuntu 23.04.

#### **4.1.2 Результати**

Під час першого етапу з блокчейн-провідників (через ендпоїнт /accounts) було отримано інформацію про облікові записи та баланс криптовалюти для кожного окремого облікового запису в мережі. Табл. 4.1 відображає загальну кількість облікових записів з додатним балансом криптовалюти, отриманих у мережі, а також статистику залишків на рахунках. Загальний баланс усіх облікових записів порахований як сума балансів кожного облікового запису мережі. Максимальне значення балансу – це найбільше значення балансу серед усіх облікових записів мережі. Середнє значення балансу – арифметичне середнє, тобто відношення загального балансу до отриманої кількості облікових записів мережі.

Медіана балансів – величина, розташована посередині відсортованого ряду балансів усіх отриманих облікових записів мережі. Баланси представлені в валюті мережі і не можуть бути прямо порівнянні між собою без попереднього конвертування.

Таблиця 4.1. Інформація про облікові записи (включаючи смарт-контракти) та статистика балансу криптовалют

Мережа	Кількість облікових записів	Баланс усіх облікових записів			
		Загальний	Максимальний	Середнє	Медіана
Acala	1224	171484729.48	168607898.02	140101.90	28.82
Aves	2352	6673004.30	1760963.98	2837.16	0.06
Energy Web	50110	74200897.13	12580691.64	1480.76	0.19
Era Swap	16121	9100000000.00	5433176313.50	564481.11	10.35
Karura	5858	1164149.56	378459.08	198.73	0.29
Kava	1581668	120557035.14	54867879.90	76.22	0.00
MCH-verse	942	447838.61	201359.23	475.41	1.45
Nahmii	698	285.31	63.60	0.41	0.01
Neat.io	787	2679038.77	1077002.95	3404.12	100.00
OASYS	24140	10012023660.41	3557500019.90	414748.29	1.00
Puppynet	2362175	249907117.84	220266275.52	105.80	0.31
Rootstock	70371	21000141.98	20996519.35	298.42	0.00
SmartBCH	32289	20999852.15	20931686.58	650.37	0.00
Xiden	7218	300006996.32	299864128.00	41563.73	0.05

Після отримання даних про облікові записи відбираються смарт-контракти для подальшої обробки. Табл. 4.2 відображає ту ж інформацію, що й попередня таблиця, але лише для підмножини облікових записів – для смарт-контрактів. Величини пораховані так само як і для табл. 4.1, але в даному випадку значення беруться лише для смарт-контрактів.

Таблиця 4.2. Інформація про смарт-контракти та статистика балансу криптовалют

Мережа	Кількість облікових записів	Баланс смарт контрактів			
		Загальний	Максимальний	Середнє	Медіана
Acala	5	723.74	705.09	144.75	2.00
Aves	20	1639250.45	1638909.31	81962.52	3.99
Energy Web	159	29609148.74	9420942.57	186221.06	0.11
Era Swap	1514	7429415264.35	5433176313.50	4907143.50	26221.50
Karura	3	4560.32	4417.80	1520.11	130.01
Kava	63	1988393.73	1915785.34	31561.81	1.09
MCH-verse	4	276566.01	201359.23	69141.50	37603.39
Nahmii	604	278.09	63.60	0.46	0.01
Neatiio	1	1.00	1.00	1.00	1.00
OASYS	20	939600812.54	912738642.13	46980040.63	111558.39
Puppynet	5	8117.42	7668.42	1623.48	200.00
Rootstock	528	2849.93	1603.85	5.40	0.00
SmartBCH	152	16215.19	15171.61	106.68	0.01
Xiden	6	13169.73	12230.20	2194.96	212.07

Наступний крок передбачає отримання байт-кодів розгорнутих смарт-контрактів з блокчейн-провідників (через ендпоїнт `/address/{address}/contracts`), а також їх початкового коду, якщо він доступний. Всі байт-коди потім декомпілюються за допомогою обраного декомпілятора. Табл. 4.3 показує кількість розгорнутих байт-кодів в кожній мережі, їхній середній розмір та кількість отриманих початкових кодів. Також вона вказує, скільки байт-кодів було успішно декомпільовано.

Середній розмір смарт-контракту порахований як відношення суми розміру усіх отриманих байт-кодів до їх кількості у даній блокчейн-мережі. Вихідні коди наявні лише для тих смарт-контрактів, які були надані блокчейн-провідником для їх верифікації. Процесу декомпіляції підлягають усі отримані байт-коди, навіть коли доступний відповідний вихідний код. Під успішно декомпільованим байт-кодом смарт-контракту розуміється успішне виконання програми-декомпілятора та не враховує якість результату декомпіляції.

Таблиця 4.3. Інформація про смарт-контракти

Мережа	Середній розмір контракту (байт)	Отримані контракти	Отримані вихідні коди	Успішно декомпільовано
Acala	5398.20	5	1	4
Aves	5495.06	17	2	14
Energy Web	3940.60	159	26	123
Era Swap	353.15	1509	1457	1255
Karura	940.33	3	1	2
Kava	9403.30	63	20	57
MCH-verse	5425.25	4	0	4
Nahmii	1854.62	604	0	513
Neatiao	6649.00	1	0	1
OASYS	1817.90	20	6	17
Puppynet	8159.78	5	0	5
Rootstock	1193.87	528	23	427
SmartBCH	10246.34	152	10	127
Xiden	2534.00	6	0	4

На наступному етапі отримується інформація з провідника (через ендпоінт /address/{address}/token-balances) про токени, які належать смарт-контрактам.



Для відображення ефективності запропонованого підходу виміряно час отримання даних для подальшого аналізу. Час отримання даних для кожного вищеприписаного етапу та загальний час наведений в табл. 4.4.

Таблиця 4.4. Час отримання даних

Мережа	Час завантаження (с)			Загально (с)
	Облікових записів	Контрактів	Токенів	
Acala	58.62	12.99	6.69	78.30
Aves	67.53	19.68	12.45	99.74
Energy Web	889.63	92.88	53.21	1035.94
Era Swap	687.97	7559.63	1453.06	9700.73
Karura	252.03	5.97	3.50	261.56
Kava	48526.23	63.46	34.23	48628.76
MCH-verse	52.05	10.95	5.86	68.85
Nahmii	18.48	426.28	295.44	740.32
Neat.io	2100.09	131.61	130.85	2362.55
OASYS	644.77	23.35	14.11	682.37
Puppynet	86831.80	23.28	3.91	86869.73
Rootstock	1254.55	563.12	166.25	1984.23
SmartBCH	868.84	134.77	96.04	1099.85
Xiden	349.87	6.48	4.26	360.64

Після отримання та обробки всіх даних проведений процес аналізу. Для демонстрації процесу було обрано випадковий смарт-контракт з набору отриманих перевірених джерельних кодів контрактів – контракт з мережі Aves з адресою *0x75fA5fecE2A9783e28856c1A7EA3Af544690ebc8* [178]. На рис. 4.1 показана частина отриманого вихідного коду [179]. На рис. 4.2 показана його відповідна версія декомпільованого байт-коду [180].

```

// function to calculate the reward
// user can't withdraw before 10000 blocks
function calculate_reward(address _staker_address) public view returns(uint256) {
    Staker memory staker = stakers[_staker_address];
    uint256 blocks_since_stake = block.number - staker.block_number;
    require(blocks_since_stake >= 30000, "You can't withdraw before 10000 blocks");
    uint256 total_pre = blocks_since_stake / 30000;
    uint256 reward = (staker.amount * percentage_per_30000_blocks * total_pre) / 100;
    return reward;
}

// calculate_reward_test
function calculate_reward_test(uint256 staked_amount, uint256 blocks_since_stake) public
    uint256 total_pre = blocks_since_stake / 30000;
    uint256 reward = (staked_amount * percentage_per_30000_blocks * total_pre) / 100;
    return reward;
}

// view function to check the balance of the contract
function check_balance() public view returns(uint256) {
    return address(this).balance;
}

// function to withdraw averseum
function withdraw() public {
    Staker storage staker = stakers[msg.sender];
    uint256 reward = calculate_reward(msg.sender);
    payable(msg.sender).transfer(reward + staker.amount);

    staker.reward += reward;
    staker.amount = 0;
    staker.block_number = 0;
}

```

Рис. 4.1. Вихідний код отриманого смарт-контракту

```

def unknown72acad56(): # not payable
    if block.number - unknown9168ae72[caller].field_256 > block.number:
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if block.number - unknown9168ae72[caller].field_256 < 30000:
        revert with 0, 'You can't withdraw before 10000 blocks'
    if unknown9168ae72[caller].field_0 and unknown00615672 != unknown9168ae72[caller].field_0
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if unknown9168ae72[caller].field_0 * unknown00615672 and block.number - unknown9168ae72[ca
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    return (unknown9168ae72[caller].field_0 * unknown00615672 * block.number - unknown9168ae72

def unknownb887634d(uint256 _param1): # not payable
    require calldata.size - 4 >= 32
    require _param1 == address(_param1)
    if block.number - unknown9168ae72[address(_param1)].field_256 > block.number:
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if block.number - unknown9168ae72[address(_param1)].field_256 < 30000:
        revert with 0, 'You can't withdraw before 10000 blocks'
    if unknown9168ae72[address(_param1)].field_0 and unknown00615672 != unknown9168ae72[addres
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if unknown9168ae72[address(_param1)].field_0 * unknown00615672 and block.number - unknown9
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    return (unknown9168ae72[address(_param1)].field_0 * unknown00615672 * block.number - unkno

def withdraw(): # not payable
    if block.number - unknown9168ae72[caller].field_256 > block.number:
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if block.number - unknown9168ae72[caller].field_256 < 30000:
        revert with 0, 'You can't withdraw before 10000 blocks'
    if unknown9168ae72[caller].field_0 and unknown00615672 != unknown9168ae72[caller].field_0
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if unknown9168ae72[caller].field_0 * unknown00615672 and block.number - unknown9168ae72[ca
        revert with Panic(17) # If an arithmetic operation results in underflow or overflow d
    if unknown9168ae72[caller].field_0 * unknown00615672 * block.number - unknown9168ae72[call

```

Рис. 4.2. Декомпільований код отриманого смарт-контракту

Перед тим, як робити ручний аналіз вихідного коду контракту, було виконано запит до помічника ChatGPT надати короткий огляд. Відповідь наведена нижче:

*«The next smart contract is called StakeAVS. It allows users to stake aves tokens and receive rewards based on the amount staked and the duration of the stake. The contract is initialized with an owner address.*

*The contract includes a struct called Staker, which stores information about each staker, including the staked amount, the block number at which the stake was made, the staker's address, and the reward earned.*

*The contract has several functions. The calculate\_reward function calculates the reward for a staker based on the number of blocks since the stake was made. The check\_balance function allows users to check the balance of the contract. The withdraw function allows stakers to withdraw their staked aves tokens and their earned rewards. The premature\_withdraw function allows stakers to withdraw their staked aves tokens before the required number of blocks have passed, but they will not receive any rewards. The stake function allows users to stake aves tokens. The check\_reward function allows users to check their earned rewards. The check\_staked function allows users to check the amount of aves tokens they have staked. The check\_block\_number function allows users to check the block number at which they made their stake. The change\_owner function allows the owner to change the owner address. The withdraw\_aves function allows the owner to withdraw aves tokens from the contract. The stop\_staking function allows the owner to stop staking, while the start\_staking function allows the owner to resume staking.*

*The contract also includes a receive function, which allows the contract to receive aves tokens as funds from stakers.»*

Переклад відповіді наведений нижче:

*«Наступний смарт-контракт називається StakeAVS. Це дозволяє користувачам робити ставки на токени Aves і отримувати винагороду залежно від суми ставки та тривалості ставки. Договір ініціюється адресою власника.*

*Контракт включає структуру під назвою Staker, яка зберігає інформацію про кожного стейкера, включаючи суму ставки, номер блоку, на який була зроблена ставка, адресу стейкера та отриману винагороду.*

*Договір виконує кілька функцій. Функція calculate\_reward обчислює винагороду для стейкера на основі кількості блоків, зроблених після того, як була зроблена ставка. Функція check\_balance дозволяє користувачам перевіряти баланс контракту. Функція вилучення дозволяє стейкерам вилучати свої токени Aves і зароблені винагороди. Функція premature\_withdraw дозволяє стейкерам вилучати свої токени Aves до того, як пройде необхідна кількість блоків, але вони не отримують жодних винагород. Функція ставки дозволяє користувачам робити ставки на токени Aves. Функція check\_reward дозволяє користувачам перевіряти свої зароблені винагороди. Функція check\_staked дозволяє користувачам перевіряти кількість токенів aves, які вони поставили. Функція check\_block\_number дозволяє користувачам перевірити номер блоку, на який вони зробили ставку. Функція change\_owner дозволяє власнику змінити адресу власника. Функція draw\_aves дозволяє власнику вилучати токени aves з контракту. Функція stop\_staking дозволяє власнику припинити ставку, тоді як функція start\_staking дозволяє власнику відновити ставку.*

*Контракт також включає функцію отримання, яка дозволяє контракту отримувати токени aves як кошти від стейкерів.»*

Наступний етап передбачає ручний аналіз. Функція calculate\_reward надає винагороду в розмірі 1 % від ставки за кожні 30000 блоків. Згідно з інформацією з провідника [164], поточний середній час блоку становить 10.3 секунди, що означає, що винагорода в 1 % надається за кожні 309000 секунд. Це означає, що протягом року (31536000 секунд) власник ставки отримає майже 100 % прибутку зі своїх інвестицій.

Ще одним важливим аспектом цього контракту є функція withdraw\_aves, яка дозволяє власнику контракту знімати всі кошти в будь-який час, тим самим створюючи ситуацію, коли всі вкладники втрачають свої інвестиції. Крім того,

якщо баланс контракту формується лише із інвестицій, то невідомо звідки контракт бере кошти, щоб покрити винагороди. Ці деталі роблять контракт дуже ризикованим для використання.

#### **4.1.3 Обговорення результатів**

Метод застосування блокчейн-провідників для отримання даних з багатьох блокчейн-мереж включає 4 етапи: вибір блокчейн-провідника; аналіз архітектури системи, до якої програмна реалізація збору даних буде інтегрована; визначення способу комунікації з блокчейн-провідником; програмна реалізація збору даних з блокчейн-провідників.

Для збору даних можливо використовувати звичайні публічні вузли та JSON-RPC інтерфейс, проте блокчейн-провідники надають значні переваги перед простим JSON-RPC інтерфейсом. Вони надають додаткову інформацію, що сприяє більш повному аналізу, особливо для розуміння смарт-контрактів в блокчейн-мережах, які досліджуються. У табл. 4.1 та табл. 4.2 зазначена загальна статистична інформація по отриманих топ-акаунтах (облікових записів) та їх балансах, яка може бути використана, наприклад, для визначення пріоритетів у подальшому ручному або автоматизованому аналізі. У табл. 4.3 представлена кількість отриманих вихідних кодів смарт контрактів для різних типів мереж. Саме вихідні коди смарт-контрактів – це та інформація, яку неможливо отримати за допомогою звичайного JSON-RPC інтерфейсу блокчейн-вузла, так як вузли не зберігають вихідні коди. Крім цього, у табл. 4.3 представлена інформація про розгорнуті смарт контракти.

Деякі провідники призначені спеціально для певних мереж [181], тоді як інші підтримують різні типи мереж [182]. Хоча провідники мають різні характеристики, функціонал та обмеження [183], у більшості випадків їх можна використовувати взаємозамінно. Серед можливих провідників, для даного дослідження був обраний Blockscout. Blockscout – це платформа, спроектована для проведення поглибленого дослідження систем блокчейну. Підтримка багатьох блокчейн-мереж, широке налаштування та відкритий вихідний код надають користувачам повний доступ до всіх даних та функцій для різних блокчейн-мереж. Таким чином, Blockscout надає

користувачам широкий спектр можливостей для глибокого дослідження та розуміння сфери блокчейн. Одна з важливих функцій полягає в тому, що Blockscout надає верифіковані вихідні коди контракту та його відповідний розгорнутий байт-код, що буде використовуватись у подальшому аналізі.

Хоча Blockscout надає API для використання сторонніми додатками [184], у багатьох випадках воно вимкнено або обмежено для загального використання. Через ці обмеження є обґрунтованим виконувати веб-скрапінг і парсинг [185]. На рис. 2.1 зображений веб-інтерфейс, головна сторінка Blockscout провідника. Саме веб-інтерфейс піддається веб-скрапінгу та парсингу. Є дві основні потенційні проблеми з таким підходом. По-перше, веб-скрапінг залежить від конкретної структури та макету веб-сайту. Будь-які зміни або оновлення дизайну веб-сайту можуть перешкодити процесу скрапінгу, що вимагає постійного обслуговування та оновлення коду скрапінгу. Другою потенційною проблемою може стати введення обмежень швидкості або блокування IP-адрес при виявленні надмірної активності скрапінгу на веб-серверах. Це може порушити процес збору даних і, можливо, призвести до проблем з доступом до серверу.

Розроблена архітектура системи (рис. 2.3) націлена на отримання даних та аналізу блокчейн-мереж, а саме, смарт-контрактів. На вхід до системи подається список мереж, а на фазу аналізу подається вся необхідна інформація для аналізу смарт-контрактів, включаючи резюме смарт-контракту, сформоване помічником ChatGPT. Ця архітектура слугує лише прикладом, як може виглядати система. Насправді, вона може бути набагато складнішою та включати ще більше компонентів для більш повноцінного аналізу, але побудова такої системи виходить за рамки даного дослідження.

На основі даної архітектури був розроблений багатопотоковий (рис. 3.3) прототип системи для проведення подальших експериментів. Хоча декомпіляція та асистент ChatGPT не є частиною отримання інформації, вони є частиною обробки інформації. Тому для демонстрації повноцінного прикладу блокчейн-аналізу, ми включили ці компоненти у дану роботу.

Щодо декомпіляції, найкращі результати надають онлайн-рішення, такі як Dedaub's EVM Bytecode Decompiler [68] і Online Solidity Decompiler [69]. Але для даного аналізу було обрано офлайн-версію, оскільки пріоритетом є швидкість отримання результатів, а не читабельність чи точність отриманого псевдокоду. Проте навіть погано декомпільований байт-код можна покращити. Для збереження загальної швидкості обробки та отримання покращених результатів декомпіляції можна використовувати онлайн-версії декомпіляторів лише для обраних контрактів.

Багатопотокова реалізація (рис. 3.3) дозволила збирати дані з кількох блокчейн систем одночасно. У табл. 4.4 представлений загальний час збору інформації для кожної досліджуваної мережі та час витрачений на кожному етапі збору: отримання інформації про облікові записи, отримання смарт-контрактів, отримання інформацію про токени, якими володіє контракт. Кількість витраченого часу на збір інформації залежить від кількості інформації у конкретній блокчейн-мережі та від обмежень сервера, з якого отримується дані. Завдяки багатопотоковій реалізації, загальний час всього процесу збору даних є найбільшим загальним часом збору даних з окремої мережі. В даному експерименті найбільше часу зайняло (табл. 4.4) отримання даних з мережі Puppynet – 86869,73 секунд, що майже 24 години. Тому увесь процес тривав до 24 годин.

Використовуючи невеликі ресурси, стало можливим зібрати необхідну інформацію, доступні вихідні коди смарт-контрактів та розгорнуті байт-коди з кількох блокчейн-мереж для подальшого аналізу. Після цього, як приклад, було проведено аналіз контракту на основі зібраних даних та вихідного коду контракту (рис. 4.1). Аналіз смарт-контракту надав інформацію про його поведінку та роль, а також ризики його використання.

Можна стверджувати, що використання блокчейн-провідників є виправданим методом для отримання даних та корисної інформації за невеликий проміжок часу, використовуючи малу кількість ресурсів. Якщо бути точним, то не потрібно орендувати потужні сервери та вручну налаштовувати вузли для кожної блокчейн мережі та синхронізувати їх протягом днів або тижнів.

У запропонованого методу є і недоліки. Якщо сервер, на якому розгорнутий провідник, є повільним або штучно обмежує швидкість, доведеться витратити більше часу на збір необхідних даних. Іноді використаний провідником блокчейн-вузол може бути не повністю синхронізований і, як наслідок, надавати неактуальні дані. Крім цього, не всі блокчейн-мережі можуть мати публічно доступний екземпляр обраного провідника.

Якщо деяка мережа з області дослідження має перелічені проблеми, може бути доцільним розгорнути локальний блокчейн-вузол та провідник. Це дозволить використовувати той самий підхід та уникнути різних реалізацій.

Основним обмеженням даного підходу є великі блокчейн-мережі, так як вони передбачають надзвичайно велику кількість даних. В даному випадку, блокчейн-провідник буде ще одним проміжним інтерфейсом, який вносить затримку. Тому, раціонально обрати інший спосіб збору даних або завантажувати лише заздалегідь обмежену частину даних.

Розвиток даного дослідження може полягати у розширенні типів публічних джерел даних, включаючи інші типи блокчейн-провідників. Це дозволить підтримувати ще більшу кількість блокчейн-мереж, швидше отримувати дані та мати запасні джерела у випадку обмежень або відключення основних. Крім того, ще одним напрямом розвитку даного дослідження може бути його застосування, а саме у безперервному скануванні та пошуку даних. Це дозволить спостерігати за визначеними параметрами багатьох блокчейн-мереж у реальному часі, і, у разі чого, отримувати сповіщення про ці зміни. При подальшому розвитку даного дослідження можливі технічні труднощі пов'язані з обмеженнями джерел інформації та серверів, на яких вони знаходяться.

## **4.2 Оцінка ефективності виявлення та ідентифікації блокчейн вузлів**

### **4.2.1 Параметри експерименту**

Експериментальна частина передбачає розгортання певної кількості вузлів у основній мережі (mainnet), які будуть використані як перевірочні, щоб програмно реалізований метод зумів знайти відповідні вузли при пошуку усіх вузлів.



Перевірочні вузли розгортаються з заданою періодичністю на хмарній платформі Vultr [177]. Розгортання машин відбувається автоматично за допомогою розробленого Python-скрипта, що використовує Vultr API [186]. Машини мають наступні характеристики, що відповідають рекомендованим [187]:

- процесор: 1 віртуальний процесор;
- оперативна пам'ять: 8 ГБ;
- накопичувач: 50 ТБ;
- операційна система: Ubuntu 22.04 LTS

#### **4.2.2 Результати**

Для перевірки ефективності запропонованого методу проводиться наступний експеримент. Програмне забезпечення розробленого методу збирає список усіх доступних вузлів блокчейн-мережі RSK. Для перевірки того, що воно виявляє усі вузли, використовуються перевірочні вузли, які додатково розгортаються на час експерименту з певним періодом та у різних фізичних локаціях. Під виявленням розуміється отримання вузла у NEIGHBORS повідомленні.

Для пошуку вузлів використовуються дві варіації розробленого методу: звичайний та оптимізований варіант. Оптимізація полягає у тому, що публічний ключ, що ідентифікує нас як вузла, обирається спеціальним способом. Приватний ключ перебирається випадково, доки не буде згенерований публічний ключ, що дасть відстань до вузла (у якого запитуються сусіди), яка буде меншою або рівною 250 при максимальній 256. Це збільшує шанси потрапити у незаповнений слот Kademlia таблиці з першої спроби, так як чим менша відстань, тим менш заповнений слот. Для обох варіантів методу запити FIND\_NODE до одного вузла надсилаються доти, доки за останніх 100 повідомлень отримується новий вузол, але не менше 1000 запитів.

Результати експерименту представлені у табл. 4.5. У експерименті використовуються 6 перевірочних вузлів. Вузли розгортаються протягом 3 годин. Програмне забезпечення виявлення вузлів запускається після розгортання усіх

перевірочних вузлів. Під час експерименту реєструється час розгортання вузла та час його виявлення розробленим методом.

Таблиця 4.5. Виявлення перевірочних вузлів

Номер вузла	Фізичне розташування вузла (місто)	Час розгортання вузла	Час виявлення програмним забезпеченням	Час виявлення програмним забезпеченням з оптимізацією
1	Сідней	10:09	13:03	13:02
2	Токіо	10:48	13:01	13:01
3	Манчестер	11:08	13:06	13:02
4	Сеул	12:00	13:03	13:02
5	Сіетл	12:12	13:22	13:01
6	Варшава	12:57	13:01	13:01

На рис. 4.3 зображений графік виявлення усіх унікальних вузлів мережі з плином часу. Оптимізація вибору публічного ключа дозволила суттєво скоротити час пошуку вузлів за рахунок відсутності очікування доступного місця у слоті таблиці сусідніх вузлів.

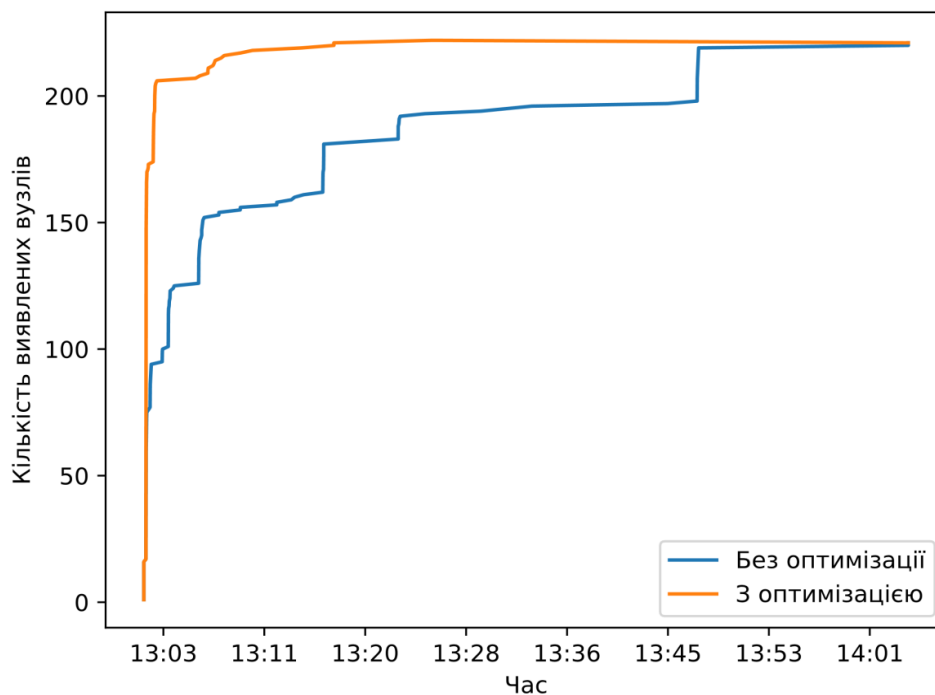


Рис. 4.3. Виявлення унікальних вузлів мережі

Загалом, розробленим методом виявлено 220 (без оптимізації) та 222 (з оптимізацією) унікальних вузлів. Під унікальними вузлами розуміється вузли які мають унікальну пару IP-адреса та ідентифікатор вузла. Тобто можлива наявність декількох вузлів з різними ідентифікаторами, але однією IP-адресою. Виявлені вузли мають 209 унікальних IP-адрес. На рис. 4.4 зображено їх фізичне розташування за допомогою сервісу [geojson.io](https://geojson.io).



Рис. 4.4. Фізичне розташування IP-адрес вузлів

Збір вузлів тривав приблизно 1 годину, а весь експеримент з розгортанням вузлів – 4 години.

#### 4.2.3 Обговорення результатів

Протоколи пошуку вузлів є основою будь-якої децентралізованої P2P мережі. Вони визначають як вузли знаходять інші вузли мережі, забезпечують її розподільність та формують певну топологію мережі. У таких системах вузли можуть під'єднуватись до кожного вузла мережі або лише до підмножини вузлів. У блокчейн-системах вузли зазвичай не зберігають інформацію про кожен вузол мережі та під'єднуються лише до невеликої підмножини вузлів для оптимізації ресурсів та ефективності комунікації.

Серед P2P систем є файлові обмінники, системи комунікації, блокчейн-мережі та інші. В залежності від функцій системи можлива різна топологія мережі та застосування різних протоколів виявлення вузлів.

Протокол виявлення вузлів в блокчейн-мережі Rootstock побудований на основі таблиці сусідніх вузлів – Kademlia, що враховує логічну відстань між вузлами. Ця відстань визначається на основі хеш-значень публічних ключів вузлів. Візуалізація розрахунку відстані між вузлами представлена на рис. 2.5. Kademlia

використовується в мережах обміну файлами та різних блокчейн-мережах. Вона підходить для децентралізованих систем, що вимагають децентралізованої топології мережі. Переваги Kademlia включають ефективність та швидкість пошуку даних, масштабованість, та низькі ресурсні витрати. До недоліків Kademlia належать проблеми безпеки (такі як вразливість до атак Sybil) та складність утримання динамічної мережі з постійно змінюваними вузлами.

Однак будь-яка документація протоколу пошуку вузлів мережі Rootstock відсутня. Хоча протокол блокчейн-мережі Ethereum теж використовує Kademlia він не зовсім ідентичний протоколу виявлення вузлів у Rootstock мережі. Тому специфікація протоколу Ethereum [152] чи існуючі дослідження не можуть бути прямо використані для побудови краулера у Rootstock мережі. Саме тому був проведений аналіз вихідного коду програмного забезпечення вузла RSKj, що дозволило отримати розуміння структури та механізмів роботи протоколу.

У мережі Rootstock протокол пошуку вузлів реалізований лише за допомогою 4 типів повідомлень (2 для запиту та 2 для відповіді). Крім того, була визначена специфічна послідовність повідомлень, яка забезпечує виявлення та з'єднання з сусідніми вузлами в мережі. Загалом, простота протоколу та реалізації є сильною стороною мережі, адже це зменшує ймовірність виникнення помилок та спрощує процес аналізу.

Так як вузли у блокчейн-мережі рівноправні, а за замовчуванням параметри Peer Discovery протоколу в RSKj однакові, то неможливо визначити критерії, за якими вузол може вважатися більш пріоритетним для надання сусідніх вузлів. При такій умові, якщо представляти мережу як граф, то він буде вважатись незваженим.

Для пошуку вузлів, а саме, обходу незваженого графа можливе використання як пошуку у глибину так і пошуку в ширину. Серед переваг використання пошуку в ширину є простота паралелізації, де кожен потік дістає наступний вузол з черги. Тому для пошуку вузлів блокчейн-мережі був обраний алгоритм пошуку у ширину.

Важливим елементом є те, що розмір слотів у таблиці Kademlia є досить невеликим. Тому досить часто можуть виникати ситуації, коли вузол не може встановити з'єднання з іншим вузлом, тому що слот відповідної відстані

заповнений. Для того щоб у вузла було більше шансів встановити з'єднання з обраним вузлом треба, щоб логічна відстань між ними була якомога меншою. А так як відстань визначається на основі ідентифікаторів вузлів (публічних ключів), то процес потрапляння вузла у вільний слот був оптимізований шляхом перебору ключів для отримання меншої відстані. Дана оптимізація відсутня у подібних дослідженнях мережі Ethereum [105] [106] [109] [188] [111] і була впроваджена у цю роботу як додатковий елемент.

Наступним важливим аспектом є кількість запитів на надання сусідів, які потрібно надіслати до одного вузла, щоб отримати його сусідів. За одне повідомлення надаються 15 найближчих сусідів та 5 випадкових. Тому надсилаючи будь-яку кількість повідомлень неможливо бути впевненим, чи отримані усі сусідні вузли або лише якась частина. Для цього використаний евристичний підхід: надсилаються повідомлення доти, доки за останніх 100 повідомлень отримується новий вузол (який ще не був виявлений у даного вузла) але у будь-якому випадку надсилається не менше 1000 запитів. Так як блокчейн-мережа динамічна, то до неї постійно підключаються та відключаються вузли. Тому опитуються також раніше опитані вузлів для повторного отримання вузлів з певною періодичністю.

При реалізації методу для пришвидшення пошуку були використані багатопотокові механізми ThreadPoolExecutor та queue мови Python. Програмне забезпечення здійснює підключення до декількох вузлів одночасно у 15 різних потоках, хоча кількість потоків може бути і більшою. Можливе навіть виділення окремого потоку для кожного вузла. Однак для уникнення потенційного блокування запитів системами запобігання вторгненню (Intrusion Prevention Systems) було вирішено обрати обмежену кількість потоків – 15, що задовольняють достатньо швидкий збір усіх вузлів.

В результаті експерименту була зібрана інформація про 222 вузли мережі, яка включає IP-адреси вузлів, UDP та TCP порти для Peer Discovery та Wire протоколів, ідентифікатори вузлів та час виявлення вузлів. Крім того, були розгорнуті перевірочні вузли, метою яких було перевірити ефективність розробленої методики виявлення. Усі 6 перевірочних вузлів були виявлені менше ніж за 10 хвилин. Дані

про час розміщення та виявлення цих тестових вузлів представлені в табл. 4.5. Експеримент також перевіряв на скільки підбір публічного ключа вузла (для пошуку меншої відстані) зменшує час пошуку вузлів. На рис. 4.3 зображений графік часу виявлення вузлів, що дозволяє впевнитись у тому, що дане покращення пришвидшує пошук. При оптимізації більшість вузлів виявлено впродовж 15 хвилин, в той час як без оптимізації для цього знадобилось трохи більше 45 хвилин. Нарешті, фізичне розташування усіх виявлених вузлів показано на рис. 4.4, що дозволяє зробити висновки про розташування вузлів. Наприклад, найбільше вузлів знаходяться у Південній Америці, Сполучених Штатах та Центральній Європі.

Можна з упевненістю стверджувати, що розроблений метод виявився високоефективним інструментом для пошуку та ідентифікації вузлів у блокчейн-мережі Rootstock. Особливістю цього методу є його здатність швидко виявляти велику кількість вузлів у мережі, де він за лічені хвилини може знайти більшість активних вузлів. Розроблений метод пошуку вузлів та відповідний експеримент дозволили отримати інформацію про топологію мережі Rootstock, яка не була представлена до цього.

Основним обмеженням розробленого методу виявлення вузлів є його нездатність ідентифікувати реальні IP-адреси вузлів, коли вони використовують анонімізуючі сервіси, такі як virtual private network (VPN) або The Onion Router (TOR). Дані сервіси призначені для забезпечення конфіденційності та анонімності користувачів шляхом приховування реальних IP-адрес за допомогою різних мережевих маршрутів та шлюзів. У результаті, коли вузол використовує VPN або TOR, метод може ідентифікувати лише IP-адресу вихідного вузла цих сервісів, а не фактичну IP-адресу вузла. Це суттєво обмежує можливості методу в контексті точного відстежування та аналізу мережевої активності.

Розвиток цього дослідження може орієнтуватися на практичне застосування розробленого методу для конкретних задач, наприклад, для аналізу блокчейн транзакцій або мережі загалом. Іншим напрямом розвитку може бути адаптація розробленого методу до інших P2P мереж, особливо тих, чії протоколи функціонують на основі механізмів, подібних до вже вивчених. Ще одним

напрямом подальшого дослідження є аналіз вузлів, які знаходяться у VPN чи TOR мережі задля отримання будь-якої інформації про реальні IP адреси вузлів.

### 4.3 Оцінка ефективності визначення відправника транзакцій за допомогою псевдовузлів

#### 4.3.1 Параметри експерименту

Для експерименту розгорнуто 10 вузлів зі стандартними налаштуваннями, які будуть транслювати транзакції до блокчейн-мережі. Версія програмного забезпечення вузла RSKj *FINGERROOT-5.2.0*. Для запуску вузла використовується наступна команда: `java -cp rskj-core/build/libs/rskj-core-*.SNAPSHOT-all.jar co.rsk.Start`. Вузли розгорнуті на платформі Vultr (рис. 4.5) та фізично розміщені у різних країнах.

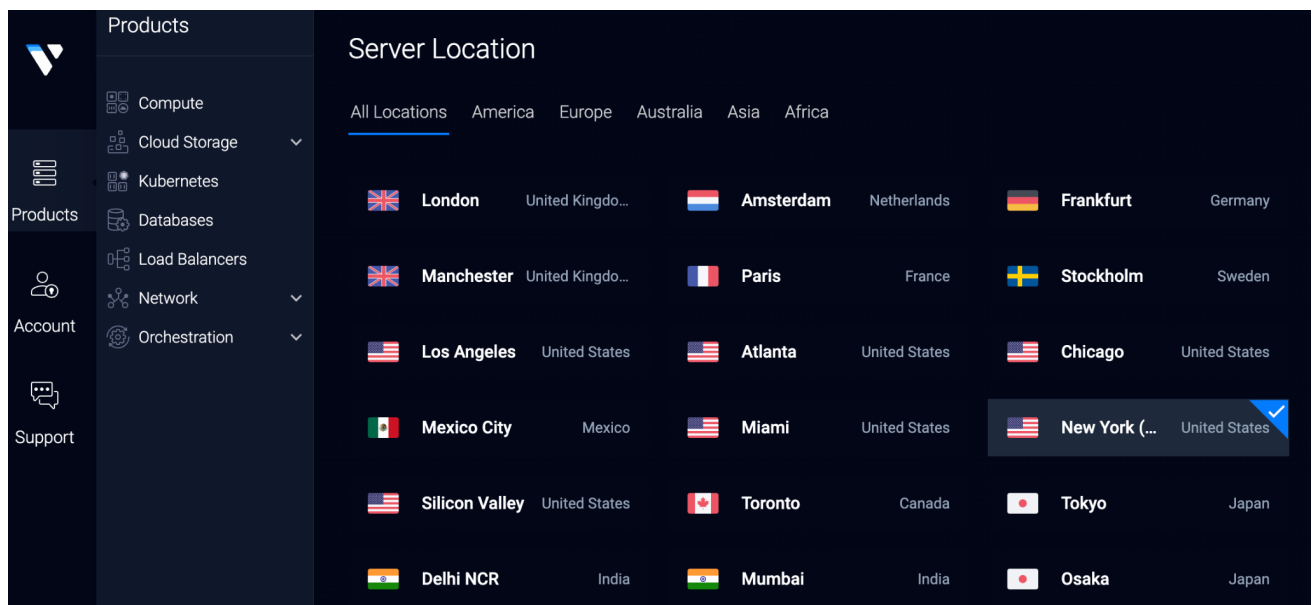


Рис. 4.5. Хмарна платформа Vultr

Програмна реалізація методу розгорнута на окремій машині. Адреси початкових вузлів взяті з конфігураційного файлу вузла: bootstrap01.rsk.co:5050, bootstrap02.rsk.co:5050, bootstrap03.rsk.co:5050, bootstrap04.rsk.co:5050, bootstrap05.rsk.co:5050, bootstrap06.rsk.co:5050, bootstrap07.rsk.co:5050, bootstrap08.rsk.co:5050, bootstrap09.rsk.co:5050, bootstrap10.rsk.co:5050,



bootstrap11.rsk.co:5050, bootstrap12.rsk.co:5050, bootstrap13.rsk.co:5050,  
bootstrap14.rsk.co:5050, bootstrap15.rsk.co:5050, bootstrap16.rsk.co:5050.

Ідея експерименту полягає у наступному:

- 10 тестових вузлів підключаються до вузлів мережі;
- програмне забезпечення методу запускається, підключається до вузлів мережі та починає реєструвати транзакції;
- з 10 тестових вузлів відправляються по 50 підписаних транзакцій з різним часовим проміжком протягом 6 годин;
- визначення першоджерела транзакції вважається успішним, якщо програмна реалізація метода отримала трансльовану транзакцію першою від першоджерела, а не від іншого вузла.

#### 4.3.2 Результати

Для відображення ефективності запропонованого підходу підраховано скільки разів відправник транзакції успішно був ідентифікований як першоджерело, скільки разів був визначений серед перших 5-ти відправників. Результати наведені в табл. 4.6.

Таблиця 4.6. Визначення першоджерела транзакцій

Тестовий вузол	Кількість відправлених транзакцій	Визначено як першоджерело	Визначено в перших 5-ти відправниках
1	50	49	50
2	50	50	50
3	50	0	0
4	50	50	50
5	50	35	37

6	50	12	12
7	50	48	48
8	50	49	49
9	50	50	50
10	50	50	50
<b>Разом</b>	<b>500</b>	<b>393</b>	<b>396</b>

Відсоток успішності для визначення першоджерела:  $393 / 500 * 100\% = 78.6\%$

Відсоток успішності для визначення в перших 5-ти відправниках:  $396 / 500 * 100\% = 79.2\%$

#### 4.3.3 Обговорення результатів

Розроблений метод є ефективним засобом для ідентифікації вузла-першоджерела транзакцій у Rootstock мережі, та у 78.6% успішно його визначає. У 79.2% першоджерело входить у топ-5 перших трансляторів транзакції. Даний метод використовує особливості реалізації Peer Discovery та Wire протоколів та потребує машини лише з однією адресою. З іншого боку, для покращення ефективності методу можуть бути використано більше псевдовузлів.

Серед обмежень даного методу є неможливість визначити IP-адресу першоджерела, якщо вузол використовує VPN або TOR. Іншим недоліком є те, що якщо у вузла перевищено ліміт підключень по Peer Discovery чи Wire протоколу, то до нього неможливо підключитись в даний момент та почати отримувати транзакції, які він транслює. Це і є основним впливом на відсоток успішності ідентифікації. Крім того, сервер, на якому розгорнутий вузол, може мати обмеження на те, у кого є доступ до підключення.

## 4.4 Оцінка ефективності аналізу OP\_RETURN-скриптів Bitcoin мережі

### 4.4.1 Параметри експерименту

Для збору OP\_RETURN-скриптів спершу проводиться повна синхронізація 834600 блоків. Позначка часу майнінгу блоку 834599 – 2024-03-14 04:11:24. У якості вузла використовується Bitcoin Core 26.0.

Параметри для методу описаного в розділі 2.4.5 обрані емпірично:

- Мінімальна довжина префіксу  $m$  – 4;
- Максимальна довжина префіксу  $n$  – 18;
- Мінімальна кількість використання префіксу  $k$  – 500.

Для вибору параметру  $q$  візуалізовано (рис. 4.6) кількість виокремлених префіксів в залежності від значення  $q$ . Було обрано таке значення ( $q = 0.17$ ), при якому кількість виокремлених префіксів суттєво зменшилась, але зберігається їх варіативність.

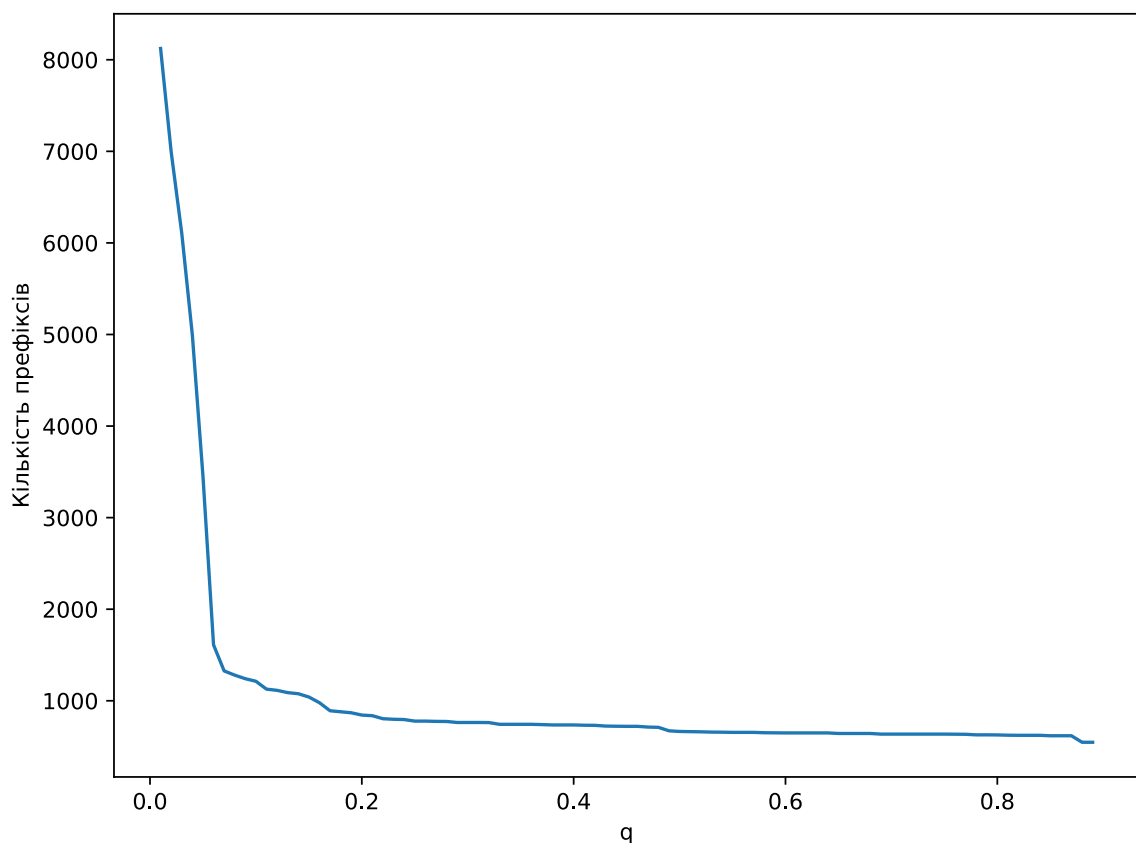


Рис. 4.6. Кількість виокремлених скриптів в залежності від значення  $q$

#### 4.4.2 Результати

В результаті експерименту отримано статистику використання різних типів вихідних скриптів. На рис. 4.7 зображений графік зміни сумарної кількості відповідних типів скриптів з кожним новим блоком. На рис. 4.8 зображений графік за логарифмічною шкалою.

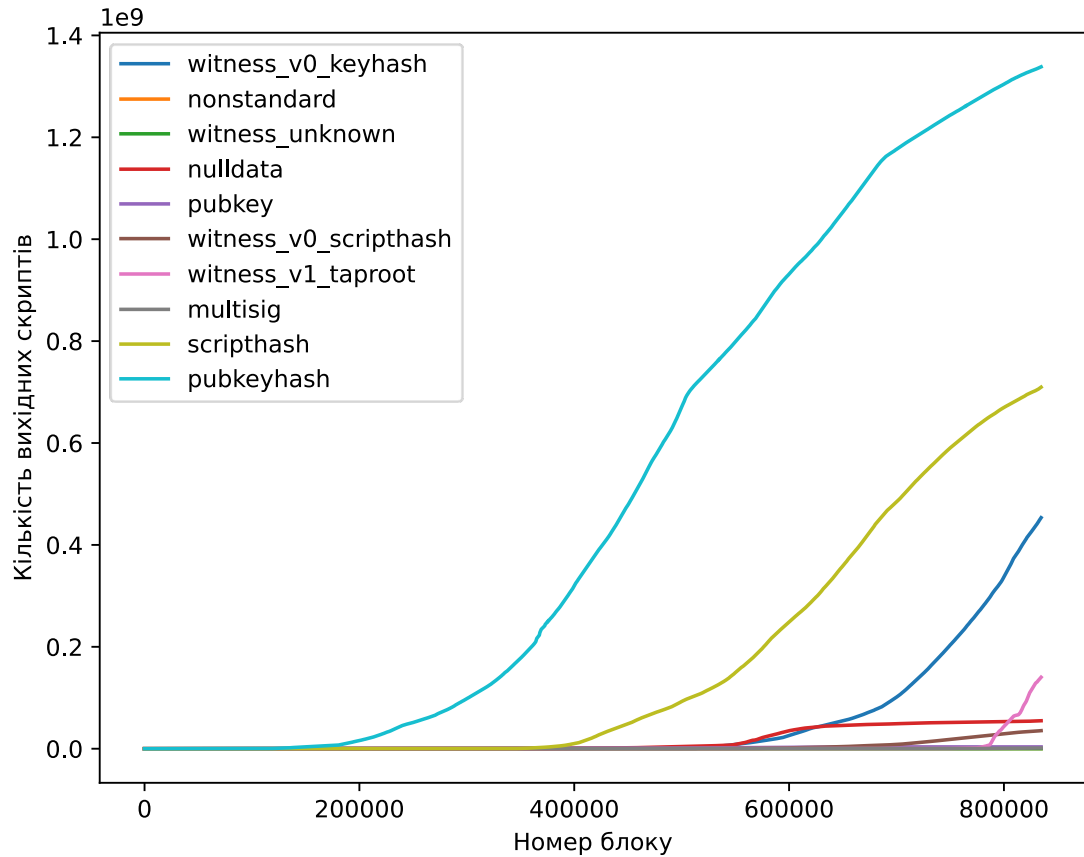


Рис. 4.7. Сумарна кількість вихідних скриптів

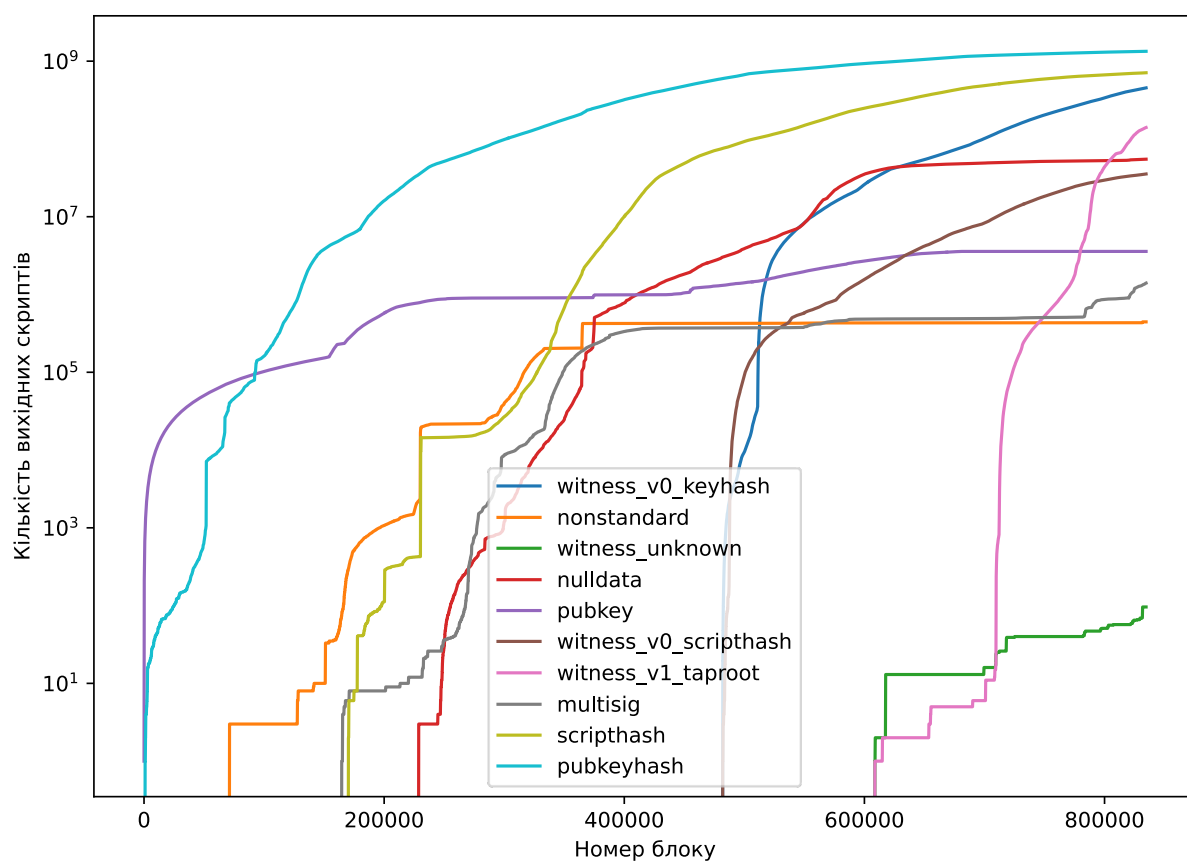


Рис. 4.8. Сумарна кількість вихідних скриптів за логарифмічною шкалою

Загальна кількість транзакцій, які містять OP\_RETURN-скрипти: 54577670.

Загальна кількість OP\_RETURN-скриптів: 54903702.

Застосовуючи алгоритм, описаний у розділі 2.4.5, виокремлено 891 префіксів, які покривають 23485521 OP\_RETURN-скриптів, тобто 42.7%. На рисунках 4.9 та 4.10 зображено частину виокремлених префіксів та кількість їх появи.

```
(b'OA\x01\x00\x04\x80', 4340)
(b'OA\x01\x00\x04\xa0', 782)
(b'OA\x01\x00\x04\xc0', 1200)
(b'OA\x01\x00\x04\xe0', 732)
(b'OA\x01\x00\x05\x80', 958)
(b'OA\x01\x00\x06', 520)
(b'OC\x00\x00\x00\x00\x00\x00', 572)
(b'OUT:', 180459)
(b'PEIRMOBILE.COM', 1444)
(b'PHOTECTOR.COM', 1538)
(b'POET\x00\x00\x00\x12 ', 14028)
(b'POR:', 856)
(b'POTX:', 1045)
(b'REFUND:', 11095)
(b'RSKBLOCK:', 156496)
(b'RSKT\x01', 721)
(b'RUNEALPHA_STAKINGc', 1214)
(b'S1I\xe6\xa37\t\x00\x00\xdb\x16\x9a0\xc8\xbaN\x8a', 2169)
(b'S1T\xa3\xbf\x7f\x1b\xaa\x90\xf8s\xf1\x91\x80\xbf\xffk\x8d', 1194)
(b'S1_\xc9\x00\x0b9\x14\x01\x82\xba\x0b9.\x8e\x05I\x97\xad\x93', 638)
(b'S1_$\xef\x08\xefa\xf9\xc8KE\xf0G\x11^\x96\x04', 928)
(b'S1~\xce4k\xa6[\x9f\xee[\xe1]}\t\x8c\xfd$b', 800)
(b'S1\x85\xe8_\xa4\xff\x05\x87\x8b\xc8\x80\x00\xce\xa0B\xab\x83', 1461)
(b'S1\x88\x13=9\xbfU\x19k\x8b!\x97\xdc2[\#\xb2', 502)
(b'S1\xad(\x17\x90 \x17G\xd3\xd3x\x02)\xfe\x9d\x05\xeb'. 712)
```

Рис. 4.9. Приклад виокремлених префіксів відсортованих в алфавітному порядку

```
(b'omni\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x00', 16562173)
(b'omni\x00\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x01', 925430)
(b'omni\x00\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x02', 371562)
(b'\xaa!\xa9\xed', 364833)
(b'\x82\x00\xbf\x80\xc7\xb5\xe3\x8a#\x86h\x00', 364308)
(b'omni\x00\x00\x00\x04\x01', 355465)
(b'\x82\x00\xbf\x80\xc7\x93\xdb\x8fR\x86h\x00', 218976)
(b'omni\x00\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x03', 190346)
(b'OUT:', 180459)
(b'RSKBLOCK:', 156496)
(b'\xb9\xe1\x1bm', 126561)
(b'\x82\x00\xbf\x80\xc7\x94\x9b\x9a\x13\x89\x80\xdd@\x00', 95503)
(b'Bitzlat0', 92115)
(b'CC\x02\x15\x00 \x12', 77653)
(b'omni\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x06', 74042)
(b'17057', 73083)
(b'=:ETH.ETH:0x', 67016)
(b'Fa\x00\x00\x00\x00', 66697)
(b'Fa\x00\x00\x00\x01', 65633)
(b'Fa\x00\x00\x00\x02', 65536)
(b'orddefi:auth', 58298)
(b'Fa\x00\x00\x00\x03', 55726)
(b'omni\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00', 46346)
(b'17056', 46283)
(b'CC\x02\x15\x00\x01', 44765)
(b'CC\x02\x01', 44419)
(b'id>>', 41081)
(b"omni\x00\x00\x00\x00\x00\x00\x00\x1f\x00\x00\x04\x8c'9", 36958)
```

Рис. 4.10. Приклад виокремлених префіксів відсортованих в порядку кількості їх появи

Серед префіксів зображених на рис. 4.10 можемо спостерігати наступні:

- *omni* префікси – відповідають протоколу Omni [189];
- *aa21a9ed* – реалізація Segregated Witness [190];
- *8200bf80c7* – не знайдено застосування, потребує аналізу;
- *4f55543a (OUT:)* – не знайдено застосування, потребує аналізу;
- *52534b424c4f434b3a (RSKBLOCK:)* – реалізація Merged Mining у мережі Rootstock [191];
- *b9e11b6d* – реалізація Merged Mining у мережі VCash [192].

#### 4.4.3 Обговорення результатів

Розроблений метод, який передбачає попередню побудову базу даних префіксів на основі кількості їх, є ефективним способом аналізу *OP\_RETURN*-скриптів.

Основним недоліком даного методу є неможливість визначити чи префікс є загальним для певної кількості префіксів чи вони є окремими типами.

Розвиток даного методу може полягати у збільшенні точності розпізнавання префіксів *OP\_RETURN*-скриптів та розпізнавання їх підтипів. Це дозволить точніше розпізнавати роль транзакції, що використовують *OP\_RETURN*-скрипти.

#### 4.5 Висновки до розділу 4

У даному розділі проведена оцінка ефективності запропонованих методів.

Ефективність методу застосування блокчейн-провідників для отримання даних для багатьох блокчейн-мереж перевірена за допомогою розробленого прототипу системи. Для заданих 14 блокчейн-мереж були зібрані дані для проведення аналізу смарт-контрактів мереж. Були зібрані дані про облікові записи (включаючи статистику по балансам), серед облікових записів виділені смарт-контракти, отримані дані про наявні токени (якими володіють смарт-контракти), зібрані байт-коди контрактів та їх вихідні коди (при наявності), проведена їх декомпіляція. Час процесу зайняв майже 24 години і для обраної віртуальної

обчислювальної машини коштував до 1 долару США. На основі зібраних даних, для прикладу повноти процесу, було проаналізовано випадковий смарт-контракт.

Ефективність запропонованого методу виявлення та ідентифікації блокчейн вузлів верифікована за допомогою розробленого прототипу програмного забезпечення та наступного експерименту у основній мережі. Розгорнуто 6 перевірочних вузлів у різних фізичних локаціях та у різний час. Період розгортання тривав до 3 годин, після чого був розпочатий пошук вузлів, який тривав 1 годину. Усі 6 перевірочних вузлів були виявлені менше ніж за 10 хвилин. За весь час пошуку розробленим методом виявлено 220 (без оптимізації) та 222 (з оптимізацією) унікальних вузлів. Оптимізація вибору публічного ключа дозволила скоротити час пошуку вузлів майже у 3 рази за рахунок відсутності очікування доступного місця у слоті таблиці сусідніх вузлів. Крім цього, за допомогою існуючих сервісів, визначено фізичне розташування усіх 209 унікальних IP-адрес вузлів та побудована відповідна мапа.

Ефективність запропонованого методу виявлення джерела транзакцій верифікована за допомогою розробленого прототипу програмного забезпечення та експерименту у основній мережі. Розгорнуто 10 перевірочних вузлів у різних фізичних локаціях та у різний час. Через дані вузли трансльовано по 50 транзакцій. За допомогою розробленого методу вдалося досягти 78.6% точності у виявлення відправника транзакції (вузла через який було відправлено транзакцію).

Ефективність аналізу OP\_RETURN-скриптів Bitcoin мережі за допомогою запропонованого методу верифікована за допомогою розробленого прототипу програмного забезпечення та експерименту у основній мережі, що дозволило отримати покриття 42.7% усіх OP\_RETURN-скриптів.



## ВИСНОВКИ

У дисертаційній роботі вирішено важливу для блокчейн індустрії науково-технічну задачу підвищення ефективності програмних засобів аналізу блокчейн транзакцій за рахунок удосконалення та розробки нових методів аналізу. Стрімкий розвиток блокчейн індустрії та широке застосування криптовалют у різних галузях робить результати дисертаційної роботи значущими.

У результаті дисертаційного дослідження отримано такі основні наукові результати:

1. Визначено основні завдання, що розв'язуються за допомогою аналізу блокчейн транзакцій. Виділено три категорії задач: асоціювання адреси з особою, аналіз потоків транзакцій та аналіз смарт контрактів. Оглянуто ключові методології та інструментарій, які застосовуються в процесі аналізу блокчейн-транзакцій. Здійснено порівняльний огляд сучасних систем, що спеціалізуються на аналізі транзакцій у блокчейні. Також висвітлено обмеження існуючих методів і виклики, пов'язані з аналізом транзакцій. Виявлено, що одним із найбільш значущих викликів є варіативність блокчейн-мереж, кожна з яких має власні унікальні характеристики та потребує окремого дослідження.

2. Розроблено метод збору даних з блокчейн мереж з використанням блокчейн-провідників, який відрізняється від традиційного підходу прямого пошуку та завантаження даних з блокчейн-вузлів. Метод дозволяє уникнути розгортання та синхронізацію вузлів, зменшуючи кількість ресурсів потрібних та отримання даних. Використання провідників не лише спрощує доступ до даних, але й забезпечує можливість отримання офчейн інформації, такої як вихідні коди смарт-контрактів, значно полегшуючи подальший аналіз смарт-контрактів та пов'язаних з ними транзакцій.

3. Розроблено метод виявлення та ідентифікації вузлів блокчейн мережі Rootstock, який уможливорює визначення вузлів мережі через послідовне виявлення та опитування кожного окремого вузла. Прості запити до вузлів для отримання інформації про їх сусідні вузли видають лише обмежений набір найближчих вузлів,

не розкриваючи інформацію про всю мережу. Втілення цієї розробки дозволило протягом однієї години ідентифікувати 222 унікальних вузли мережі, що включають 209 унікальних IP-адрес, демонструючи високу ефективність методу у зборі даних про структуру мережі.

4. Розроблено метод встановлення вихідного вузла транзакції в блокчейн мережі Rootstock. Факт отримання транзакції від конкретного вузла не є безпосереднім показником того, що цей вузол є початковим джерелом її відправлення. Запропонований метод продемонстрував здатність з точністю у 78.6% випадків коректно ідентифікувати вузол, з якого була трансльована транзакція, що свідчить про високий рівень ефективності даного підходу в процесі аналізу транзакцій.

5. Удосконалено методологію аналізу OP\_RETURN-скриптів Bitcoin мережі. Розроблений метод ґрунтується на аналізі префіксної частини скриптів, та дозволяє ідентифікувати транзакції, які використовують опкод OP\_RETURN для побудови різноманітних протоколів, що дозволило покрити 42.7% усіх OP\_RETURN-скриптів. Особливістю даного підходу є здатність виявляти раніше невідомі префіксні частини, шляхом аналізу частоти їх появи в скриптах. Це, в свою чергу, дозволяє розпізнавати нові протоколи. Такий підхід розширює можливості аналітичного дослідження використання скриптів OP\_RETURN;

6. Створено комплексну архітектуру для системи аналізу транзакцій у блокчейн-мережах, яка складається з чотирьох основних рівнів: інтерфейс користувача, який включає графічний та консольний варіанти; Web API, що забезпечує взаємодію між інтерфейсом користувача та ядром системи; ядро системи, яке координує всю взаємодію в межах системи; та агенти, задіяні для збору, обробки інформації та взаємодії з різними блокчейн-мережами. Використання модульної структури через агентів надає системі високу гнучкість та масштабованість, дозволяючи легко інтегрувати нові блокчейн-мережі та розширювати функціонал системи. Розроблені в рамках цього дисертаційного дослідження методи інтегровані в систему шляхом введення нових агентів або

адаптації вже існуючих, що сприяє посиленню ефективності аналітичної платформи.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- [1] Y. Dorogyu та V. Kolisnichenko, «Blockchain Transaction Analysis: A Comprehensive Review of Applications, Tasks and Methods,» *System Research & Information Technologies*, № 4, pp. 37-53, doi: 10.20535/SRIT.2308-8893.2023.4.03, 2023.
- [2] Y. Dorogyu та V. Kolisnichenko, «Devising a method for rapid data retrieval using explorers for blockchain analysis,» *Eastern-European Journal of Enterprise Technologies*, т. 4, № 2, pp. 6-16, doi: <https://doi.org/10.15587/1729-4061.2023.286079>, 2023.
- [3] Я. Дорогий та В. Колісніченко, «Застосування логуювання різними учасниками блокчейн-мереж для деанонімізації кінцевого користувача,» *Вісник Хмельницького національного університету Серія: «Технічні науки»*, т. 1, № 5, pp. doi: <https://www.doi.org/10.31891/2307-5732-2023-325-5-60-66>, 2023.
- [4] Y. Dorogyu та V. Kolisnichenko, «Developing a method for the detection and identification of rootstock blockchain network nodes,» *Eastern-European Journal of Enterprise Technologies*, т. 1, № 2, pp. 6-15, doi: <https://doi.org/10.15587/1729-4061.2024.297903>, 2024.
- [5] V. Buterin, «Ethereum: A next-generation smart contract and decentralized application platform,» [Онлайновий]. Available: [https://finpedia.vn/wp-content/uploads/2022/02/Ethereum\\_white\\_paper-a\\_next\\_generation\\_smart\\_contract\\_and\\_decentralized\\_application\\_platform-vitalik-buterin.pdf](https://finpedia.vn/wp-content/uploads/2022/02/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf).
- [6] Chainalysis, «The Crypto Crime Report,» February 2023. [Онлайновий]. Available: [https://go.chainalysis.com/rs/503-FAP-074/images/Crypto\\_Crime\\_Report\\_2023.pdf](https://go.chainalysis.com/rs/503-FAP-074/images/Crypto_Crime_Report_2023.pdf).

- [7] «Blockchain Security and AML Analysis Report - 2022 Annual,» [Онлайновий]. Available: [https://www.slowmist.com/report/2022-Blockchain-Security-and-AML-Analysis-Annual-Report\(EN\).pdf](https://www.slowmist.com/report/2022-Blockchain-Security-and-AML-Analysis-Annual-Report(EN).pdf).
- [8] L. Prinsloo та R. Henderson, «Trail of Brothers Linked to Missing Bitcoin Stash Is Still Murky,» [Онлайновий]. Available: <https://www.bloomberg.com/news/articles/2021-06-27/trail-of-brothers-linked-to-missing-bitcoin-stash-is-still-murky>.
- [9] L. Prinsloo, «Crypto Losses Probed by South African Cops After Brothers Vanish,» [Онлайновий]. Available: <https://www.bloomberg.com/news/articles/2022-01-11/crypto-losses-probed-by-south-african-cops-after-brothers-vanish>.
- [10] «Top five most wanted crypto criminals,» [Онлайновий]. Available: <https://www.cnbctv18.com/cryptocurrency/top-five-most-wanted-crypto-criminals-15897891.htm>.
- [11] «The 10 biggest crypto scams on record and the lessons we can learn from them,» [Онлайновий]. Available: <https://irishtechnews.ie/10biggestcryptoscams/>.
- [12] «Protect your privacy,» [Онлайновий]. Available: <https://bitcoin.org/en/protect-your-privacy>.
- [13] «Ukrainian Cyber Police Department In Collaboration with Crystal,» [Онлайновий]. Available: <https://crystalintelligence.com/crystal-news/ukrainian-cyber-police-department-now-in-collaboration-with-crystal-blockchain/>.
- [14] K. Baker, «What is Ransomware as a Service (RaaS)?,» 2023. [Онлайновий]. Available: <https://www.crowdstrike.com/cybersecurity-101/ransomware/ransomware-as-a-service-raas/>.
- [15] «Chainalysis in Action: U.S. Authorities Disrupt NetWalker Ransomware,» [Онлайновий]. Available: <https://www.chainalysis.com/blog/netwalker-ransomware-disruption-arrest/>. [Дата звернення: March 2024].

- [16] «Rekt - Leaderboard,» [Онлайновый]. Available: <https://rekt.news/leaderboard/>.
- [17] «SlowMist Hacked - SlowMist Zone,» [Онлайновый]. Available: <https://hacked.slowmist.io/>.
- [18] «Кросчейн мосты,» [Онлайновый]. Available: <https://academy.binance.com/uk/glossary/cross-chain-bridges>.
- [19] «The 10 Biggest Crypto Exchange Hacks In History,» [Онлайновый]. Available: <https://crystalintelligence.com/articles/the-10-biggest-crypto-exchange-hacks-in-history/>.
- [20] @officer\_cia, «How Cross-Chain Bridges are Hacked?,» [Онлайновый]. Available: <https://officercia.mirror.xyz/AFkEUuxid1egNm4XdqYEzWEwosPNbz2CNgh1NrQ7LZQ>.
- [21] @officer\_cia, «Retrospective: hacks in web3,» [Онлайновый]. Available: <https://telegra.ph/Retrospective-hacks-in-web3-10-24>.
- [22] «SlowMist AML: Tracking funds laundered by Tornado Cash,» [Онлайновый]. Available: <https://slowmist.medium.com/slowmist-aml-tracking-funds-laundered-by-tornado-cash-3a0e1f637054>.
- [23] «Elliptic Analysis: New York Husband and Wife Arrested for Allegedly Laundering \$5 Billion in Bitcoin Stolen from Bitfinex in 2016,» [Онлайновый]. Available: <https://www.elliptic.co/blog/elliptic-analysis-new-york-husband-and-wife-arrested-for-laundering-5-billion-in-bitcoin-stolen-from-bitfinex-in-2016>.
- [24] «Crystal Blockchain Analytics: Investigation of the Zaif Exchange Hack,» [Онлайновый]. Available: [https://bitfury.com/content/downloads/bitfury\\_crystal\\_zarif\\_report\\_23\\_10\\_18.pdf](https://bitfury.com/content/downloads/bitfury_crystal_zarif_report_23_10_18.pdf).
- [25] «DeviantArt Protect: Helping Safeguard Your Art,» DeviantArt, [Онлайновый]. Available: <https://www.deviantart.com/team/journal/DeviantArt-Protect-Helping-Safeguard-Your-Art-884278903>.

- [26] «Cryptocurrency Regulations Around The World,» ComplyAdvantage, [Онлайновый]. Available: <https://complyadvantage.com/insights/cryptocurrency-regulations-around-world/>.
- [27] T. Akhtar та S. Shukla, «China Makes a Comeback in Bitcoin Mining Despite Government Ban,» [Онлайновый]. Available: <https://www.bloomberg.com/news/articles/2022-05-17/china-makes-a-comeback-in-bitcoin-mining-despite-government-ban>.
- [28] «Cryptocurrency transaction monitoring: What you need to know,» ComplyAdvantage, [Онлайновый]. Available: <https://complyadvantage.com/insights/transaction-monitoring-cryptocurrencies/>.
- [29] «How Continuous Cryptocurrency Transaction Monitoring Gives Compliance Teams Peace of Mind,» [Онлайновый]. Available: <https://blog.chainalysis.com/reports/kyt-continuous-monitoring/>.
- [30] «Cryptocurrency Regulation: How Governments Around the World Regulate Crypto,» [Онлайновый]. Available: <https://www.chainalysis.com/blog/cryptocurrency-regulation-explained/>.
- [31] «Financial Intermediaries: their role on real examples,» [Онлайновый]. Available: <https://agicap.com/en/article/financial-intermediaries/>.
- [32] M. Morel, «Technical Analysis Is Dead, Long Live Transaction Analysis,» [Онлайновый]. Available: <https://www.nasdaq.com/articles/technical-analysis-is-dead-long-live-transaction-analysis>.
- [33] U.S. Department of the Treasury, «Illicit Finance Risk Assessment of Decentralized Finance,» April 2023. [Онлайновый]. Available: <https://home.treasury.gov/system/files/136/DeFi-Risk-Full-Review.pdf>.
- [34] «Cryptocurrency: Risk management overview,» Willis Towers Watson, [Онлайновый]. Available: <https://www.wtwco.com/-/media/WTW/Insights/2019/01/cryptocurrency-risk-management-overview.pdf>.

- [35] «Blockchain risk management,» [Онлайновый]. Available: <https://www2.deloitte.com/content/dam/Deloitte/us/Documents/financial-services/us-fsi-blockchain-risk-management.pdf>.
- [36] M. White, «Web3 is Going Just Great,» [Онлайновый]. Available: <https://web3isgoinggreat.com/>.
- [37] S. Kessler та D. Nelson, «Polygon Blockchain Nodes Briefly Went Out of Sync, Affecting Explorer, Sowing Confusion,» [Онлайновый]. Available: <https://www.coindesk.com/tech/2023/02/22/polygon-blockchain-suffers-apparent-outage/>.
- [38] T. Chang, J. Ho, Z. Tirrell, G. Weng та J. You, «A Risk Classification Framework for Decentralized Finance Protocols,» 2022. [Онлайновый]. Available: <https://dr.soa.org/4aa5bb/globalassets/assets/files/resources/research-report/2022/decentralized-finance-protocols.pdf>.
- [39] V. Gaur та A. Gaiha, «Building a Transparent Supply Chain,» [Онлайновый]. Available: <https://hbr.org/2020/05/building-a-transparent-supply-chain>.
- [40] P. Dutta, T.-M. Choi, S. Somani та R. Butala, «Blockchain technology in supply chain operations: Applications, challenges and research opportunities,» *Transportation Research Part E: Logistics and Transportation Review*, т. 142, p. doi: <https://doi.org/10.1016/j.tre.2020.102067>, 2020.
- [41] Intuz , «How AI And Blockchain Technologies Are Reforming Supply Chain Management,» [Онлайновый]. Available: <https://www.intuz.com/blog/ai-and-blockchain-in-supply-chain-management>.
- [42] L. Compagnucci, D. Lepore, F. Spigarelli, E. Frontoni, M. Baldi та L. D. Berardino, «Uncovering the potential of blockchain in the agri-food supply chain: An interdisciplinary case study,» *Journal of Engineering and Technology Management*, т. 65, p. doi: <https://doi.org/10.1016/j.jengtecman.2022.101700>, 2022.



- [43] «Incident report: Rootstock peg-out service outage (Fixed),» [Онлайновый]. Available: <https://blog.rsk.co/noticia/incident-report-rsk-peg-out-service-outage/>.
- [44] «Data and Analytics,» ethereum.org, [Онлайновый]. Available: <https://ethereum.org/en/developers/docs/data-and-analytics/>.
- [45] «Network Congestion,» [Онлайновый]. Available: <https://learn.bybit.com/glossary/definition-network-congestion/>.
- [46] «Scaling,» [Онлайновый]. Available: <https://ethereum.org/en/developers/docs/scaling/>. [Дата звернення: 24 April 2023].
- [47] S. D. Lerner, «RSK Scalability,» [Онлайновый]. Available: <https://medium.com/iovlabs-innovation-stories/rsk-scalability-c44252f05a4b>.
- [48] «State of research: increasing censorship resistance of transactions under proposer/builder separation (PBS),» [Онлайновый]. Available: [https://notes.ethereum.org/@vbuterin/pbs\\_censorship\\_resistance](https://notes.ethereum.org/@vbuterin/pbs_censorship_resistance).
- [49] R. Behnke, «How blockchain DDoS attacks work,» [Онлайновый]. Available: <https://www.halborn.com/blog/post/how-blockchain-ddos-attacks-work>.
- [50] «Forta: a decentralized runtime security solution for automated threat detection and prevention on smart contracts,» [Онлайновый]. Available: <https://docs.forta.network/en/latest/2022-7-11%20Forta%20Litepaper.pdf>.
- [51] «Forta Solutions in the Wild,» [Онлайновый]. Available: <https://forta.notion.site/Forta-Solutions-in-the-Wild-71e63d933ef5426d92642a8019708d48>.
- [52] «Leaked Slides Show How Chainalysis Flags Crypto Suspects for Cops,» CoinDesk, [Онлайновый]. Available: <https://www.coindesk.com/business/2021/09/21/leaked-slides-show-how-chainalysis-flags-crypto-suspects-for-cops/>.

- [53] «Bitcoin abuse database,» [Онлайновий]. Available: <https://www.bitcoinabuse.com/>. [Дата звернення: 24 April 2023].
- [54] «Ethereum (ETH) Blockchain Explorer,» [Онлайновий]. Available: <https://etherscan.io/>.
- [55] N. Tovanich, N. Heulot, J.-D. Fekete та P. Isenberg, «Visualization of Blockchain Data: A Systematic Review,» *IEEE Transactions on Visualization and Computer Graphics*, т. 27, № 7, 2021.
- [56] J. S. Tharani, E. Y. A. Charles, Z. Hou, M. Palaniswami та V. Muthukkumarasamy, «Graph Based Visualisation Techniques for Analysis of Blockchain Transactions,» в *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, 2021.
- [57] «The ultimate guide to graph visualization,» [Онлайновий]. Available: <https://info.cambridge-intelligence.com/graph-visualization-white-paper>.
- [58] "Crystal," [Online]. Available: <https://explorer.crystalblockchain.com/>. [Accessed 24 April 2023].
- [59] «Introducing Cross-Chain Investigations in Reactor: Enhancing Crypto Tracing,» [Онлайновий]. Available: <https://www.chainalysis.com/blog/cross-chain-investigations/>.
- [60] «How cryptomixers allow cybercriminals to clean their ransoms,» [Онлайновий]. Available: <https://intel471.com/blog/cryptomixers-ransomware>.
- [61] «Blockchain bridges: An industry overview,» [Онлайновий]. Available: <https://blog.rsk.co/noticia/blockchain-bridges-an-industry-overview/>. [Дата звернення: 2 December 2023].
- [62] T. Tironsakkul, M. Maarek, A. Eross та M. Just, «Tracking Mixed Bitcoins,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2009.14007>, 2020.
- [63] M. J. Shayegan та H. R. Sabor, «A Collective Anomaly Detection Method Over Bitcoin Network,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2107.00925>, 2021.

- [64] Z. Wu, J. Liu, J. Wu та Z. Zheng, «TRacer: Scalable Graph-based Transaction Tracing for Account-based Blockchain Trading Systems,» в *IEEE Transactions on Information Forensics and Security*, 2022.
- [65] J. Siegenthaler, *Blockchain Clustering with Machine Learning*, 2020.
- [66] «Top 32 blockchain analysis tools,» [Онлайновый]. Available: <https://startupstash.com/blockchain-analysis-tools/>.
- [67] «Verifying smart contracts,» [Онлайновый]. Available: <https://ethereum.org/en/developers/docs/smart-contracts/verifying/>.
- [68] «Bytecode Decompilation,» [Онлайновый]. Available: <https://library.dedaud.com/decompile>. [Дата звернення: 24 April 2023].
- [69] «Online Solidity Decompiler,» [Онлайновый]. Available: <https://ethervm.io/decompile/>.
- [70] «OpenChain Monorepo,» [Онлайновый]. Available: <https://github.com/openchainxyz/openchain-monorepo>.
- [71] @w1nt3r\_eth, «A list of power tools that security researchers use to investigate hacks,» [Онлайновый]. Available: [https://twitter.com/w1nt3r\\_eth/status/1597998923226177543](https://twitter.com/w1nt3r_eth/status/1597998923226177543).
- [72] «EVM Tracing,» [Онлайновый]. Available: <https://geth.ethereum.org/docs/developers/evm-tracing>.
- [73] «Announcing our fully featured, portable solidity debugger,» [Онлайновый]. Available: <https://archive.trufflesuite.com/blog/announcing-full-portable-solidity-debugger/>.
- [74] «Static and Dynamic Analysis - Ethereum Smart Contract Best Practices,» ConsenSys Diligence, [Онлайновый]. Available: <https://consensys.github.io/smart-contract-best-practices/security-tools/static-and-dynamic-analysis/>.

- [75] J. Feist, G. Grieco та A. Groce, «Slither: A Static Analysis Framework for Smart Contracts,» в *2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain*, 2019.
- [76] «Fuzzing,» ConsenSys Diligence, [Онлайновий]. Available: <https://consensys.io/diligence/fuzzing/>.
- [77] G. Grieco, W. Song, A. Cygan, J. Feist та A. Groce, «Echidna: effective, usable, and fast fuzzing for smart contracts,» в *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2020.
- [78] M. Mossberg, F. Manzano, E. Hennenfent, A. Groce, G. Grieco, J. Feist, T. Brunson та A. Dinaburg, «Manticore: A User-Friendly Symbolic Execution Framework for Binaries and Smart Contracts,» в *2019 34th IEEE/ACM International Conference on Automated Software Engineering*, 2019 .
- [79] «MythX: Preparing for a smart contract audit,» [Онлайновий]. Available: <https://mythx.io/about/>.
- [80] «GPT-4,» [Онлайновий]. Available: <https://openai.com/gpt-4>.
- [81] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro та Y. Zhang, «Sparks of Artificial General Intelligence: Early experiments with GPT-4,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2303.12712>, 2023.
- [82] «Codex (and GPT-4) can't beat humans on smart contract audits,» Trail of Bits, [Онлайновий]. Available: <https://blog.trailofbits.com/2023/03/22/codex-and-gpt4-cant-beat-humans-on-smart-contract-audits/>.
- [83] «Anonymity Services' Usage of Cryptocurrency and Role in Cybercrime,» [Онлайновий]. Available: <https://www.chainalysis.com/blog/anonymity-services-cryptocurrency/>.
- [84] «AMLBot - The full-fledged crypto compliance solution,» [Онлайновий]. Available: <https://amlbot.com/>.

- [85] R. Werner, S. Lawrenz та A. Rausch, «Blockchain Analysis Tool of a Cryptocurrency,» в *Proceedings of the 2020 The 2nd International Conference on Blockchain Technology*, 2020.
- [86] «Hardware requirements,» [Онлайновий]. Available: <https://geth.ethereum.org/docs/getting-started/hardware-requirements>.
- [87] Z. Luo, R. Murukutla та A. Kate, «Last Mile of Blockchains: RPC and Node-as-a-service,» *arXiv*, p. doi: <https://arxiv.org/abs/2212.03383>, 2022.
- [88] H. Kalodner, M. Möser, K. Lee, S. Goldfeder, M. Plattner, A. Chator та A. Narayanan, «BlockSci: Design and applications of a blockchain analysis platform,» в *29th USENIX Security Symposium*, 2020.
- [89] H. Kuzuno та C. Karam, «Blockchain explorer: An analytical process and investigation environment for bitcoin,» в *APWG Symposium on Electronic Crime Research*, 2017.
- [90] X. Wen, K. S. Yeo, Y. Wang, L. Cheng, F. Zhu та M. Zhu, «Code Will Tell: Visual Identification of Ponzi Schemes on Ethereum,» в *CHI Conference on Human Factors in Computing Systems*, 2023.
- [91] M. Dotan, Y.-A. Pignolet, S. Schmid, S. Tochner та A. Zohar, «Survey on Cryptocurrency Networking: Context, State-of-the-Art, Challenges,» *arXiv*, p. doi: <https://doi.org/10.48550/ARXIV.2008.08412>, 2020.
- [92] S. T.-B. Yahya Hassanzadeh-Nazarabadi, S. Otoum, S. Ucar та Ö. Özkasap, «DHT-based Communications Survey: Architectures and Use Cases,» *arXiv*, p. doi: <https://doi.org/10.48550/ARXIV.2109.10787>, 2021.
- [93] «Sync modes,» [Онлайновий]. Available: <https://geth.ethereum.org/docs/fundamentals/sync-modes>. [Дата звернення: 15 August 2023].
- [94] «go-ethereum - Official Go implementation of the Ethereum protocol,» [Онлайновий]. Available: <https://geth.ethereum.org>.

- [95] «go-ethereum - bootnodes.go,» [Онлайновый]. Available: <https://github.com/ethereum/go-ethereum/blob/67979022aa8ea6a96b618c086f60b49af7d2a568/params/bootnodes.go#L23>.
- [96] «Ethereum Foundation,» [Онлайновый]. Available: <https://ethereum.org/en/foundation/>.
- [97] C. Grunspan та R. Pérez-Marco, «The mathematics of Bitcoin,» *arXiv*, p. doi: <https://arxiv.org/abs/2003.00001>, 2020.
- [98] «miner Namespace,» [Онлайновый]. Available: <https://geth.ethereum.org/docs/interacting-with-geth/rpc/ns-miner>.
- [99] «How to Send Private Transactions on Ethereum,» [Онлайновый]. Available: <https://www.alchemy.com/overviews/ethereum-private-transactions>.
- [100] N. M. Al-Fannah, «One leak will sink a ship: WebRTC IP address leaks,» в *2017 International Carnahan Conference on Security Technology (ICCST)*, 2017.
- [101] Y. Lu та G. Tsudik, «Towards Plugging Privacy Leaks in the Domain Name System,» в *IEEE Tenth International Conference on Peer-to-Peer Computing*, 2010.
- [102] «Browserleaks - Check your browser for privacy leaks,» [Онлайновый]. Available: <https://browserleaks.com/>. [Дата звернення: 15 August 2023].
- [103] «Etherscan - Crunchbase Company Profile,» [Онлайновый]. Available: <https://www.crunchbase.com/organization/etherscan>.
- [104] A. Howell, T. Saber та M. Bendeche, «Measuring node decentralisation in blockchain peer to peer networks,» *Blockchain: Research and Applications*, т. 4, № 1, 2023.
- [105] J.-P. Eisenbarth, T. Cholez та O. Perrin, «Ethereum's Peer-to-Peer Network Monitoring and Sybil Attack Prevention,» *Journal of Network and Systems Management*, 2022.

- [106] S. Henningsen, D. Teunis, M. Florian та B. Scheuermann, «Eclipsing Ethereum Peers with False Friends,» *arXiv*, p. doi: <https://doi.org/10.48550/ARXIV.1908.10141>, 2019.
- [107] V. Deshpande, H. Badis та L. George, «BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology,» в *2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*, 2018.
- [108] D. Xu, J. Gao, L. Zhu, F. Gao та J. Zhao, «Statistical and clustering analysis of attributes of Bitcoin backbone nodes,» *PLoS ONE*, № 18, p. doi: <https://doi.org/10.1371/journal.pone.0292841>, 2023.
- [109] K. Li, Y. Tang, J. Chen, Y. Wang та X. Liu, «TopoShot: uncovering Ethereum's network topology leveraging replacement transactions,» в *Proceedings of the 21st ACM Internet Measurement Conference*, 2021.
- [110] A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring та B. Bhattacharjee, «Discovering Bitcoin's Public Topology and Influential Nodes,» 2015.
- [111] E. M. Eyong, *Analyzing the Peer-to-Peer Network of Ethereum*, 2019.
- [112] C. Z. Taotao Wang, Q. Yang, S. Zhang та S. C. Liew, «Ethna: Analyzing the Underlying Peer-to-Peer Network of Ethereum Blockchain,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2010.01373>, 2020.
- [113] A. Biryukov та S. Tikhomirov, «Deanonymization and Linkability of Cryptocurrency Transactions Based on Network Analysis,» в *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019.
- [114] G. Fanti та P. Viswanath, «Deanonymization in the bitcoin P2P network,» в *Advances in Neural Information Processing Systems 30*, 2017.
- [115] M. Apostolaki, C. Maire та L. Vanbever, «PERIMETER: A Network-Layer Attack on the Anonymity of Cryptocurrencies,» в *Financial Cryptography and Data Security*, 2021.

- [116] I. D. Mastan та S. Paul, «A New Approach to Deanonymization of Unreachable Bitcoin Nodes,» в *Cryptology and Network Security*, Springer International Publishing, 2018.
- [117] M. Bartoletti, S. Lande, L. Pompianu та A. Bracciali, «A general framework for blockchain analytics,» в *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, 2017.
- [118] QuantaBytes, «A Survey of Bitcoin Transaction Types,» [Онлайновий]. Available: <https://www.quantabytes.com/articles/a-survey-of-bitcoin-transaction-types>.
- [119] «WeirdSignature.xlsx,» [Онлайновий]. Available: <https://drive.google.com/file/d/0BwdyTvSh6bUkeUM5QzRXRWkteWM/edit?resourcekey=0-vNzKij15GrcDAA24-jyKHA>.
- [120] M. Bartoletti та L. Pompianu, «An Analysis of Bitcoin OP\_RETURN Metadata,» в *Financial Cryptography and Data Security. Lecture Notes in Computer Science*, 2017, pp. 218–230, doi: 10.1007/978-3-319-70278-0\_14.
- [121] «Cool data embedded in the Bitcoin blockchain (Ciro's Bitcoin Inscription Museum),» [Онлайновий]. Available: <https://ourbigbook.com/cirosantilli/cool-data-embedded-in-the-bitcoin-blockchain>.
- [122] M. Bartoletti, B. Bellomy та L. Pompianu, «A Journey into Bitcoin Metadata,» *Journal of Grid Computing*, т. 17, pp. 3-22, doi: 10.1007/s10723-019-09473-3, 2019.
- [123] «Bitcoin non-standard transaction,» [Онлайновий]. Available: <https://ourbigbook.com/cirosantilli/bitcoin-non-standard-transaction>.
- [124] S. Bistarelli, I. Mercanti та F. Santini, «An Analysis of Non-standard Transactions,» *Frontiers in Blockchain*, т. 2, p. doi: 10.3389/fbloc.2019.00007, 2019.
- [125] «Cryptocurrency Prices, Charts And Market Capitalizations,» [Онлайновий]. Available: <https://coinmarketcap.com/>.



- [126] S. Sharma та M. Naggar, «A New Era for Bitcoin?,» 2023. [Онлайновый]. Available: <https://research.binance.com/static/pdf/a-new-era-for-bitcoin.pdf>.
- [127] «Ethereum - Verified Contracts,» [Онлайновый]. Available: <https://etherscan.io/contractsVerified/>.
- [128] «Blockchair - Universal blockchain explorer and search engine,» [Онлайновый]. Available: <https://blockchair.com>.
- [129] «Chains & projects using blockscout,» [Онлайновый]. Available: <https://docs.blockscout.com/about/projects>.
- [130] «Blockchain Explorer - Bitcoin Tracker & More,» [Онлайновый]. Available: <https://www.blockchain.com/explorer>.
- [131] «Blockchain Explorer By Bitquery. Bitquery Explorer.,» [Онлайновый]. Available: <https://explorer.bitquery.io/>.
- [132] «Etherscan Explorer Services,» [Онлайновый]. Available: <https://etherscan.io/eaas>.
- [133] «Unmarshal Blockchain Explorer,» [Онлайновый]. Available: <https://xscan.io/>.
- [134] «Open Source Ethereum Explorer,» [Онлайновый]. Available: <https://beaconcha.in>.
- [135] «otterscan: A blazingly fast, local, Ethereum block explorer built on top of Erigon,» [Онлайновый]. Available: <https://github.com/otterscan/otterscan>.
- [136] «Blockhead - track, visualize & explore all of crypto, DeFi & web3,» [Онлайновый]. Available: <https://blockhead.info/explorer>.
- [137] «Ethereal: Eternal is a block explorer for EVM-based chains,» [Онлайновый]. Available: <https://github.com/tryeternal/eternal>.
- [138] «3xpl,» [Онлайновый]. Available: <https://3xpl.com/>.
- [139] «ethVM: An Open Source Block Explorer for Ethereum with Users In Mind,» [Онлайновый]. Available: <https://github.com/EthVM/EthVM>.
- [140] «explorer: Block explorer showcasing the BlockCypher APIs,» [Онлайновый]. Available: <https://github.com/blockcypher/explorer>.

- [141] «Search for block, transaction, address,» [Онлайновый]. Available: <https://blockexplorer.one/>.
- [142] «The General Multi-chain Explorer and Blockchain API,» [Онлайновый]. Available: <https://tokenview.io/>.
- [143] «BSCTrace - BNB Smart Chain (BSC) Blockchain Explorer,» [Онлайновый]. Available: <https://blockchain.coinmarketcap.com>.
- [144] OpenAI, «GPT-4 Technical Report,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2303.08774>, 2023.
- [145] «OpenAI Platform,» [Онлайновый]. Available: <https://platform.openai.com/docs/api-reference>.
- [146] P. Maymounkov та D. Mazières, «Kademlia: A Peer-to-Peer Information System Based on the XOR Metric,» в *Peer-to-Peer Systems (IPTPS 2002)*, 2002.
- [147] «rskj - PeerExplorer.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/FINGERROOT-5.2.0/rskj-core/src/main/java/co/rsk/net/discovery/PeerExplorer.java>.
- [148] «rskj - NodeDistanceTable.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/FINGERROOT-5.2.0/rskj-core/src/main/java/co/rsk/net/discovery/table/NodeDistanceTable.java>.
- [149] «rskj - KademliaOptions.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/FINGERROOT-5.2.0/rskj-core/src/main/java/co/rsk/net/discovery/table/KademliaOptions.java>.
- [150] «rskj - ECKey.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/ff1b67ee2c3a2158f212a199f71f3a3ad1918965/rskj-core/src/main/java/org/ethereum/crypto/ECKey.java#L287>.
- [151] «rskj - PeerDiscoveryMessageFactory.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/FINGERROOT-5.2.0/rskj-core/src/main/java/co/rsk/net/discovery/message/PeerDiscoveryMessageFactory.java>.

- [152] «Node Discovery Protocol,» [Онлайновый]. Available: <https://github.com/ethereum/devp2p/blob/master/discv4.md>.
- [153] «rskj - PeerExplorer.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/ff1b67ee2c3a2158f212a199f71f3a3ad1918965/rskj-core/src/main/java/co/rsk/net/discovery/PeerExplorer.java#L221>.
- [154] «rskj - PeerExplorer.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/ff1b67ee2c3a2158f212a199f71f3a3ad1918965/rskj-core/src/main/java/co/rsk/net/discovery/PeerExplorer.java#L319>.
- [155] «ECRecover and Signature Verification in Ethereum,» [Онлайновый]. Available: <https://coders-errand.com/ecrecover-signature-verification-ethereum/>.
- [156] «rskj - reference.conf,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/e0e81174e3154760c3a41af38cef622481ddb192/rskj-core/src/main/resources/reference.conf#L131>.
- [157] «rskj - RskWireProtocol.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/e0e81174e3154760c3a41af38cef622481ddb192/rskj-core/src/main/java/co/rsk/net/eth/RskWireProtocol.java#L130>.
- [158] «rskj - MessageType.java,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/FINGERROOT-5.1.0/rskj-core/src/main/java/co/rsk/net/messages/MessageType.java>.
- [159] «Beautifulsoup4 - PyPI,» [Онлайновый]. Available: <https://pypi.org/project/beautifulsoup4/>.
- [160] «panoramix: Ethereum decompiler,» [Онлайновый]. Available: <https://github.com/palkeo/panoramix>.
- [161] S. Meisami та W. E. Bodell, «A Comprehensive Survey of Upgradeable Smart Contract Patterns,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2304.03405>, 2023.

- [162] «multiprocessing - Process-based parallelism,» [Онлайновый]. Available: <https://docs.python.org/3/library/multiprocessing.html>.
- [163] «Acala chain explorer,» [Онлайновый]. Available: <https://blockscout.acala.network>.
- [164] «Aves Explorer,» [Онлайновый]. Available: <https://avescan.io>.
- [165] «Energy Web Chain Energy Web Foundation Explorer,» [Онлайновый]. Available: <https://explorer.energyweb.org>.
- [166] «Eraswap Explorer,» [Онлайновый]. Available: <https://eraswap.info>.
- [167] «Karura Chain Explorer,» [Онлайновый]. Available: <https://blockscout.karura.network>.
- [168] «Kava Ethereum Co-Chain Explorer,» [Онлайновый]. Available: <https://explorer.kava.io>.
- [169] «MCH Verse Explorer,» [Онлайновый]. Available: <https://explorer.oasys.mycryptoheroes.net>.
- [170] «Nahmii Explorer,» [Онлайновый]. Available: <https://explorer.nahmii.io>.
- [171] «Neatio NIO Explorer,» [Онлайновый]. Available: <https://scan.neatio.net>.
- [172] «Oasys Explorer,» [Онлайновый]. Available: <https://scan.oasys.games>.
- [173] «BONE BONE explorer,» [Онлайновый]. Available: <https://puppyscan.shib.io>.
- [174] «Rootstock (RBTC) explorer,» [Онлайновый]. Available: <https://blockscout.com/rsk/mainnet>. [Дата звернення: 14 December 2023].
- [175] «SmartBCH explorer,» [Онлайновый]. Available: <https://sonar.cash>. [Дата звернення: 7 April 2023].
- [176] «Xiden Explorer,» [Онлайновый]. Available: <https://explorer.xiden.com>.
- [177] «SSD VPS Servers, Cloud Servers and Cloud Hosting,» [Онлайновый]. Available: <https://www.vultr.com>.
- [178] «StakeAVS (0x75fA5fecE2A9783e28856c1A7EA3Af544690ebc8) - Explorer,» [Онлайновый]. Available:

<https://avescan.io/address/0x75fA5fecE2A9783e28856c1A7EA3Af544690ebc8>  
.

- [179] «Stake.sol,» [Онлайновый]. Available: <https://gist.github.com/VaWheel/072250e3b419fb9ad8e5c9b411776579>.
- [180] StakeDecompiled.py, <https://gist.github.com/VaWheel/ad80df2f8b15067388876e92d8b80901>.
- [181] «RSK explorer,» [Онлайновый]. Available: <https://explorer.rsk.co>. [Дата звернення: 14 December 2023].
- [182] «Blockchair - Crunchbase Company Profile & Funding,» [Онлайновый]. Available: <https://www.crunchbase.com/organization/blockchair>.
- [183] «Block Explorers,» [Онлайновый]. Available: <https://www.alchemy.com/best/block-explorers>.
- [184] «API - Blockscout,» [Онлайновый]. Available: <https://docs.blockscout.com/for-users/api>.
- [185] A. Brenning та S. Henn, «Web scraping: a promising tool for geographic data acquisition,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2305.19893>, 2023.
- [186] «Vultr API – Vultr.com,» [Онлайновый]. Available: <https://www.vultr.com/api/>.
- [187] «Hardware requirements,» [Онлайновый]. Available: <https://dev.rootstock.io/rsk/node/install/requirements/>.
- [188] N. Zarin, I. Sheff та S. Roos, «Blockchain Nodes are Heterogeneous and Your P2P Overlay Should be Too: PODS,» *arXiv*, p. doi: <https://doi.org/10.48550/arXiv.2306.16153>, 2023.
- [189] «spec - OmniSpecification.adoc,» [Онлайновый]. Available: <https://github.com/OmniLayer/spec/blob/master/OmniSpecification.adoc>.
- [190] «bips - bip-0141.mediawiki,» [Онлайновый]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki#commitment-structure>.

- [191] «RSK: Bitcoin Merge Mining is Here to Stay,» [Онлайновый]. Available: <https://blog.rootstock.io/noticia/rsk-bitcoin-merge-mining-is-here-to-stay/>.
- [192] «vcash - pool\_merge\_mining.md,» [Онлайновый]. Available: [https://github.com/vcashorg/vcash/blob/vcash/doc/pool\\_merge\\_mining.md#what-do-you-need-to-do](https://github.com/vcashorg/vcash/blob/vcash/doc/pool_merge_mining.md#what-do-you-need-to-do).
- [193] «rskj-main.conf,» [Онлайновый]. Available: <https://github.com/rksmart/rskj/blob/ff1b67ee2c3a2158f212a199f71f3a3ad1918965/rskj-core/src/main/resources/config/main.conf#L32>. [Дата звернення: 14 December 2023].

# ДОДАТОК А

DocuSign Envelope ID: C2B5566D-49D3-4D15-8D23-80EE4BA72BA6



March 8th, 2024

## Present

**Ref.:** Act on Dissertation Results

To whom it may concern

Reference is made to the results of the dissertation research done by Kolisnichenko Vadym, a postgraduate of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

We hereby inform you that, some of the results from the dissertation in "Methods and software tools for analysis of blockchain transactions" made by Kolisnichenko Vadym, are used in the day-to-day work of IOV Labs.

In particular, IOV Labs uses and implements the following results from Kolisnichenko Vadym research:

1. Development of a method for detection and identification of nodes of the Rootstock blockchain network, which involves sending specially formed messages to nodes to obtain their neighbors.
2. The graph search algorithm is used to traverse all received neighboring nodes. This allows IOV to consistently detect all network nodes.
3. The solution helps to monitor and analyze the blockchain network in continuous mode.

Yours sincerely,

DocuSigned by:  
*Matthias Rosenthal*  
08E905D9AB474AB

Matthias Rosenthal  
IOV Labs Limited  
Suite 23 Portland House, Glacis Road GI  
Legal Representative

DS  
AT

## ДОДАТОК Б

### Список публікацій здобувача

*Наукові праці, в яких опубліковані основні наукові результати дисертації:*

1. Dorogyu, Y., Kolisnichenko, V. Blockchain Transaction Analysis: A Comprehensive Review of Applications, Tasks and Methods. *System research and information technologies*. 2023. Vol. 4. P. 37–53. (Scopus Q4).
2. Dorogyu, Y., Kolisnichenko, V. Devising a method for rapid data retrieval using explorers for blockchain analysis. *Eastern-European Journal of Enterprise Technologies*. 2023. Vol. 4, No. 2 (124). P. 6–16. (Scopus Q3, фахове видання категорії «А»).
3. Дорогий Я. Ю., Колісніченко В.Ю. Застосування логування різними учасниками блокчейн-мереж для деанонімізації кінцевого користувача. *Вісник Хмельницького національного університету. Серія: «Технічні науки»*. 2023. Т. 1, №5. С. 60–66. (Фахове видання категорії «Б»).
4. Dorogyu, Y., Kolisnichenko, V. Developing a method for the detection and identification of rootstock blockchain network nodes. *Eastern-European Journal of Enterprise Technologies*. 2024. Vol. 1, No. 2 (127). P. 6–15. (Scopus Q3, фахове видання категорії «А»).