Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

В.о. зав. кафедрою КІСМіТ
Марина ЄСІНА
«Допущено до захисту»

« »_____2025р.

Пояснювальна записка
до кваліфікаційної роботи бакалавра
на тему: «Дослідження та аналіз можливостей технології блокчейн для розподіленого зберігання та передачі криптографічних ключів»

оцінка «_____ »
Голова ЕК
Мичуда Л.З.

Керівник: PhD Вілігура В. В.
Рецензент: к.т.н. проф. Качко О. Г.
Виконавець: студент групи КБ-42
Курдюмов М. О.

Харків 2025

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалавра містить 59 сторінок, 4 рисунки, 5 таблиць, 19 посилань на джерела.

Метою роботи є дослідження та аналіз можливостей використання технології блокчейн для організації розподіленого зберігання і передачі криптографічних ключів, а також, розробка програмного прототипу відповідної системи та проведення його тестування.

Об’єктом дослідження є процес децентралізованого зберігання даних із використанням технології блокчейн.

Предметом дослідження є методи та засоби розподіленого зберігання і передачі криптографічних ключів на основі блокчейн-систем. Основними методами дослідження є:

Аналіз наукової літератури та існуючих технічних рішень у сфері блокчейн-технологій і криптографії.

Системний аналіз та моделювання процесів розподіленого зберігання даних у блокчейн-мережах.

Криптографічний аналіз методів захисту та передачі ключів (симетричне та асиметричне шифрування, цифровий підпис, PKI).

Проєктування архітектури і програмна реалізація прототипу системи для зберігання та передачі ключів.

Експериментальне тестування прототипу та оцінка його працездатності (аналіз продуктивності, безпеки і надійності).

У вступній частині роботи наведено огляд основних понять блокчейн-технологій, принципів побудови розподілених реєстрів і алгоритмів консенсусу. Детально розглядаються характеристики різних типів блокчейнів (відкритих, особистих, консорціальних) та їх ключові властивості (незмінність записів, децентралізація, стійкість до відмов). Окрему увагу приділено огляду методів

криптографічного захисту даних: пояснено принципи симетричного й асиметричного шифрування, створення та управління відкритим і особистими ключами, роль системи відкритих ключів (PKI) і цифрових підписів у забезпеченні цілісності й автентифікації. У наступних ах проведено аналіз існуючих підходів до розподіленого зберігання та передачі криптографічних ключів у блокчейн-мережах. Розглянуто моделі з використанням смарт-контрактів, протоколів розподіленого генерування ключів та механізмів голосування для оновлення ключів. Проаналізовано переваги таких рішень (прозорість операцій, захист від фальсифікації, відсутність єдиної точки відмови) та виявлено обмеження (вплив швидкості консенсусу на продуктивність, питання масштабування і конфіденційності). Практична частина роботи присвячена проєктуванню та реалізації програмного прототипу системи розподіленого зберігання криптографічних ключів на базі блокчейна. Описано вибір технологій (наприклад, використання платформи Ethereum чи Hyperledger для розгортання блокчейн-мережі) та архітектуру рішення. Наведено алгоритми створення та реєстрації ключових пар, запису гешів ключів у блокчейн, а також процедури відновлення і передачі ключів між учасниками. Демонструється, як смарт-контракти забезпечують автоматизацію процесів управління ключами та зберігають історію їх змін у незмінному реєстрі. Наступний розділ містить результати експериментального тестування розробленого прототипу. Прототип перевірено в умовах реального навантаження з емульованою мережею вузлів. Оцінено показники продуктивності (час затвердження транзакцій, витрати обчислювальних ресурсів) та безпеки (стійкість до спроб несанкціонованого доступу чи модифікації даних). Результати тестування підтвердили ефективність запропонованого підходу: система забезпечує надійне зберігання й передачу криптографічних ключів у децентралізованому середовищі без значних втрат продуктивності. На основі отриманих даних сформульовано рекомендації щодо оптимізації та подальшого розвитку системи.

Можливості використання результатів у практичній діяльності дуже різносторонні.

Кібербезпека: отримані результати можна застосувати для створення децентралізованих сховищ ключів у системах захисту інформації, що підвищує стійкість до зовнішніх атак і вразливостей центральних серверів.

Системи відкритих ключів (PKI): запропонований підхід може бути використаний у модернізації PKI, де блокчейн забезпечує прозоре ведення реєстру сертифікатів і ключів без необхідності в довірених третіх сторонах.

Цифрова інфраструктура: розроблене рішення доцільно впроваджувати в проэктах з побудови надійної цифрової інфраструктури (громадські реєстри, управління ідентифікацією), де потрібен децентралізований контроль доступу та автентифікації.

Ключові слова: БЛОКЧЕЙН, РОЗПОДІЛЕНЕ ЗБЕРІГАННЯ ДАНИХ, КРИПТОГРАФІЧНІ КЛЮЧІ, ДЕЦЕНТРАЛІЗАЦІЯ, КОНСЕНСУСНІ АЛГОРИТМИ, СМАРТ-КОНТРАКТИ, ОСОБИСТИЙ КЛЮЧ, ВІДКРИТИЙ КЛЮЧ, СИМЕТРИЧНЕ ШИФРУВАННЯ, АСИМЕТРИЧНЕ ШИФРУВАННЯ, СИСТЕМА ВІДКРИТИХ КЛЮЧІВ, ІНФОРМАЦІЙНА БЕЗПЕКА, ЕЛЕКТРОННИЙ ПІДПИС, РОЗПОДІЛЕНА МЕРЕЖА.

ABSTRACT

The explanatory note to the bachelor's qualification thesis contains 59 pages, 4 figures, 5 tables, and 19 references.

The aim of the work is to research and analyze the possibilities of using blockchain technology for organizing distributed storage and transmission of cryptographic keys, as well as to develop a software prototype of the corresponding system and conduct its testing.

The object of the research is the process of decentralized data storage using blockchain technology.

The subject of the research is the methods and means of distributed storage and transmission of cryptographic keys based on blockchain systems.

The main research methods include:

Analysis of scientific literature and existing technical solutions in the fields of blockchain and cryptography.

System analysis and modeling of distributed data storage processes in blockchain networks.

Cryptographic analysis of key protection and distribution methods (symmetric and asymmetric encryption, digital signatures, PKI).

Architectural design and software implementation of a prototype system for key storage and transfer.

Experimental testing of the prototype and evaluation of its performance (productivity, security, and reliability analysis).

The introductory section of the thesis presents an overview of basic concepts in blockchain technology, principles of distributed ledger construction, and consensus algorithms. It explores various types of blockchains (public, private, consortium) and their key properties (immutability, decentralization, fault tolerance). Special attention is paid to cryptographic data protection methods, including principles of symmetric and

asymmetric encryption, public and private key management, the role of Public Key Infrastructure (PKI), and digital signatures in ensuring data integrity and authentication.

Subsequent sections analyze existing approaches to distributed storage and transfer of cryptographic keys in blockchain networks. Models using smart contracts, distributed key generation protocols, and key rotation mechanisms are reviewed. Advantages such as transparency, tamper resistance, and elimination of single points of failure are analyzed, along with identified limitations like the impact of consensus speed on performance, scalability challenges, and confidentiality issues.

The practical part of the work focuses on designing and implementing a software prototype of a blockchain-based cryptographic key management system. It describes the choice of technologies (e.g., Ethereum or Hyperledger for deploying the blockchain network) and the architecture of the solution. Algorithms for key pair creation and registration, hash storage in the blockchain, and key recovery and transfer procedures are detailed. The system demonstrates how smart contracts automate key management and maintain a tamper-proof record of changes.

The next section presents the results of experimental testing. The prototype was tested under realistic load conditions in a simulated node network. Performance indicators (transaction confirmation time, computational resource consumption) and security indicators (resistance to unauthorized access or data modification) were evaluated. The results confirmed the effectiveness of the proposed approach: the system provides reliable key storage and transmission in a decentralized environment without significant performance degradation. Based on the findings, recommendations for optimization and further development were proposed.

The results of this research have wide practical applicability:

Cybersecurity: The findings can be used to build decentralized key storage systems that enhance resilience against external attacks and single points of failure.

Public Key Infrastructure (PKI): The proposed approach can modernize PKI systems by allowing transparent key and certificate management without the need for trusted third parties.

Digital Infrastructure: The developed solution is applicable to projects aimed at creating robust digital infrastructures (public registries, identity management), where decentralized access control and authentication are essential.

Keywords: BLOCKCHAIN, DISTRIBUTED DATA STORAGE, CRYPTOGRAPHIC KEYS, DECENTRALIZATION, CONSENSUS ALGORITHMS, SMART CONTRACTS, PRIVATE KEY, PUBLIC KEY, SYMMETRIC ENCRYPTION, ASYMMETRIC ENCRYPTION, PUBLIC KEY INFRASTRUCTURE, INFORMATION SECURITY, DIGITAL SIGNATURE, DISTRIBUTED NETWORK.

ЗМІСТ

ПЕРЕЛІК ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

NCDP – нейроколородинамічне програмування зображень

PKI – інфраструктура відкритих ключів (Public Key Infrastructure)

СКК – система управління ключами (Key Management System)

HSM – апаратний модуль безпеки (Hardware Security Module)

CA – центр сертифікації (Certificate Authority)

RA – реєстраційний центр (Registration Authority)

CRL – список відкликаних сертифікатів (Certificate Revocation List)

OCSP – протокол перевірки статусу сертифікатів онлайн (Online Certificate Status Protocol)

CP – політика сертифікації (Certificate Policy)

CPS – опис практик сертифікації (Certification Practice Statement)

RFC – запит на коментар (Request for Comments)

AES – стандарт симетричного шифрування (Advanced Encryption Standard)

MIT – Массачусетський технологічний інститут (Massachusetts Institute of Technology)

API – інтерфейс прикладного програмування (Application Programming Interface)

IoT – інтернет речей (Internet of Things)

TLS – протокол захисту транспортного рівня (Transport Layer Security)

SSL – протокол захищених сокетів (Secure Sockets Layer)

S/MIME – безпечне розширення для електронної пошти (Secure/Multipurpose Internet Mail Extensions)

SCPKI – децентралізована система управління сертифікатами (Smart Contract-based Public Key Infrastructure)

ВСТУП

Сучасні інформаційні технології дедалі глибше інтегруються в усі сфери діяльності людини, що водночас породжує зростаючі вимоги до забезпечення конфіденційності, цілісності та доступності інформації. Одним із ключових компонентів систем захисту даних є криптографічні ключі, які використовуються для шифрування, автентифікації та електронного підпису. Від надійності зберігання та управління цими ключами значною мірою залежить ефективність та безпечність усієї інформаційної інфраструктури.

Традиційні моделі зберігання криптографічних ключів зазвичай ґрунтуються на централізованих рішеннях, таких як сховища ключів або сертифікаційні центри. Водночас така централізація створює потенційно небезпечні точки відмови, які можуть стати об’єктом атак або джерелом внутрішніх порушень безпеки. У зв’язку з цим виникає потреба у дослідженні альтернативних, більш стійких до загроз підходів до розподілу та захисту криптографічної інформації.

Одним із таких перспективних напрямів є використання технології блокчейн – децентралізованої структури, що дозволяє забезпечити незмінність, прозорість і доступність даних без потреби у централізованих посередниках. У контексті управління криптографічними ключами блокчейн може виступати як механізм збереження гешів відкритих ключів, реєстрації операцій передачі або відкликання сертифікатів, а також забезпечення контрольованого доступу до ключової інформації через смарт-контракти.

Актуальність обраної теми зумовлена як стрімким розвитком блокчейн-технологій, так і зростанням потреби у більш безпечних, надійних та децентралізованих рішеннях у сфері управління ключами. Крім того, інтеграція блокчейну у криптографічну інфраструктуру відкриває нові можливості для вдосконалення систем PKI, IoT-безпеки, міжмережевого шифрування та цифрової ідентифікації.

Мета дипломної роботи полягає у дослідженні можливостей використання технології блокчейн для розподіленого зберігання та безпечної передачі криптографічних ключів, а також у розробці і тестуванні програмного прототипу відповідної системи. У роботі проаналізовано існуючі підходи, розглянуто типові проблеми безпеки, змодельовано архітектуру системи та проведено її експериментальну перевірку.

Практична значущість дослідження полягає в можливості впровадження розробленого підходу у сферах кібербезпеки, побудови систем PKI нового покоління, інфраструктури електронного врядування, корпоративних систем управління доступом і цифрової автентифікації.

1 ТЕОРЕТИЧНІ ОСНОВИ УПРАВІЛННЯ КРИПТОГРАФІЧНИМИ КЛЮЧАМИ ТА ТЕХНОЛОГІЇ БЛОКЧЕЙН

1.1 Типи, життєвий цикл та управління криптографіічними ключами

Сучасні криптографічні системи базуються на використанні криптографічних ключів – секретних параметрів, від захищеності яких залежить безпека всієї системи [2]. Безпека інформації, що захищається криптографічно, прямо залежить від надійності ключів та ефективності заходів з їхнього захисту [1]. Наприклад, у стандарті NIST SP 800-57 ключ порівнюється з кодом від сейфа: якщо зловмиснику відомий цей код, то навіть найнадійніший сейф не захистить вміст [1]. Відтак криптографічні ключі є критично важливим елементом системи захисту, і компрометація ключа зводить нанівець дію навіть стійких алгоритмів шифрування. Існують два основних типи криптографічних ключів: симетричні та асиметричні. Симетричний (секретний) ключ – це єдиний спільний ключ, який відомий учасникам сеансу і використовується для шифрування та дешифрування даних у симетричних алгоритмах (наприклад, AES) [1]. Той самий симетричний ключ застосовується для обох операцій, тому його необхідно зберігати в таємниці, доступній тільки уповноваженим сторонам [1]. Натомість асиметрична криптографія оперує парою ключів – відкритим і закритим (особистим) ключем, що математично пов’язані між собою [1]. Відкритий ключ може вільно розповсюджуватися і використовуватися для шифрування даних або перевірки цифрового підпису, тоді як відповідний закритий ключ зберігається в секреті його власником і використовується для дешифрування або створення підпису [1]. Криптосистема побудована таким чином, що дані, зашифровані відкритим ключем, може розшифрувати лише відповідний особистий ключ, і навпаки; аналогічно, цифровий підпис, сформований особистим ключем, перевіряється тільки парним відкритим ключем. Вибір типу ключа визначає методи його розповсюдження і

використання. Симетричні ключі потребують конфіденційного розподілу між сторонами – ключ має бути переданий по захищеному каналу або встановлений за допомогою спеціального протоколу обміну ключами. Існує так звана «проблема початкового обміну»: для шифрування потрібен спільний секретний ключ, але щоб безпечно передати цей ключ, вже потрібен захищений канал, який, у свою чергу, можна забезпечити лише за наявності ключа. Для розв’язання цього замкненого кола зазвичай використовується попередній обмін ключами іншими засобами (наприклад, вручення носія з ключем особисто) або використання асиметричних протоколів узгодження сеансового ключа [3]. В асиметричних системах відкриті ключі можуть вільно поширюватися (зокрема, через сертифікати у інфраструктурі відкритих ключів, PKI), однак виникає інше завдання – забезпечення автентичності відкритого ключа та довіри до нього. Таким чином, обидва типи ключів є невід’ємними компонентами сучасної криптографії: симетричні ключі зазвичай застосовуються для швидкого шифрування великих обсягів даних, тоді як асиметричні спрощують розподіл секретної інформації і використовуються для захисту симетричних ключів, цифрових підписів тощо.

1.2 Управління ключами: політики, стандарти та ризики

Ефективне управління криптографічними ключами в організації потребує впровадження формалізованих політик і використання спеціалізованих засобів. Політика управління ключами являє собою високорівневий документ, що визначає загальні принципи, правила та відповідальність щодо поводження з ключами в організації [4]. Така політика окреслює, хто відповідає за генерацію та розподіл ключів, як визначаються криптоперіоди і довжина ключів, якими стандартами керуються при цьому, які застосовуються механізми контролю (аудит, розподіл ролей), а також порядок дій у разі інцидентів з ключами (компрометація, відмова в роботі тощо) [4].

На основі політики розробляються детальні процедури (Key Management Practices Statement), де описується організаційна структура управління ключами, конкретні операційні процедури та відповідальні особи за виконання функцій, визначених політикою [4]. Для реалізації цих політик на практиці, особливо в середовищах з великою кількістю користувачів і ключів, застосовуються системи управління ключами. СКК – це програмно-апаратні рішення, призначені для централізованого створення, зберігання, розподілу і обслуговування ключів та пов’язаних з ними метаданих [4].

Автоматизована система управління ключами може виконувати такі завдання, як генерація криптопараметрів, розподіл ключів між компонентами, регулярна ротація та відстеження строків дії ключів, резервне копіювання, відновлення, моніторинг використання ключів і реєстрація (логування) операцій з ними [4, 5]. Використання централізованого СКК підвищує безпеку (за рахунок єдиних стандартів і контролю доступу) та спрощує адміністрування, знижуючи ймовірність помилок людини. В новітньому огляді [5] відзначається тенденція розвитку різних архітектур СКК (локальні, хмарні, гібридні), і запропоновано їх класифікацію за вимогами алгоритмів, стадіями життєвого циклу ключів та сферами застосування. Зокрема, сучасні рішення пропонують інтеграцію зі схемами розподіленого управління (напр. блокчейн для децентралізованого зберігання секретів), підтримку апаратних модулів безпеки, інтерфейси для розробників (API) тощо [5]. Вибір конкретної системи управління ключами залежить від потреб організації щодо масштабованості, рівня безпеки, відповідності нормативним вимогам та бюджету. Дотримання стандартів відіграє ключову роль у забезпеченні належного управління ключами. Одним з базових документів галузі є рекомендація NIST SP 800-57 Part 1 Rev. 5 (2020), яка містить узагальнені принципи і найкращі практики управління криптографічними ключами [4]. Цей стандарт класифікує типи ключів та визначає вимоги до їх захисту, окреслює можливі стани життєвого циклу ключа (наприклад, активний, призупинений, скомпрометований,

деактивований, знищений), описує функції, що входять до процесу управління ключами, та розглядає низку питань, пов’язаних з ключовим матеріалом (використання та довжина ключів, криптоперіоди, контроль параметрів алгоритмів, управління сертифікатами, аудит, відновлення після компрометації тощо) [4].

Існують також міжнародні стандарти, такі як ISO/IEC 11770 (Key Management Techniques) та галузеві норми (наприклад, ANSI X9.17 для фінансових установ), що регламентують процедури управління ключами [6]. Дотримання цих стандартів та рекомендацій забезпечує сумісність рішень та належний рівень захисту на всіх етапах роботи з ключами. Неналежне управління ключами створює серйозні вразливості для інформаційної безпеки. Зокрема, компрометація секретного або особистого ключа одразу призводить до втрати конфіденційності даних: якщо нападник отримав ключ шифрування, він зможе розшифрувати всю інформацію, захищену цим ключем, або видати себе за законного користувача, оперуючи вкраденим особистим ключем (що дозволяє підробляти електронні підписи тощо) [1].

1.3 Традиційні методи розподілу і зберігання ключів

У криптографічних системах надзвичайно важливим є належний розподіл і безпечне зберігання ключів шифрування. Традиційно застосовувалися різноманітні методи управління ключами – від фізичного (ручного) обміну секретними ключами між сторонами до впровадження централізованих служб автентифікації та інфраструктури відкритих ключів (PKI). Далі розглянуто основні традиційні підходи до розповсюдження ключової інформації. Додатково висвітлено роль цифрових сертифікатів, центрів сертифікації, політик сертифікації та списків відкликаних сертифікатів у межах PKI, а насамкінець – проаналізовано обмеження і проблеми традиційних схем розподілу ключів (питання довіри, масштабованості та адміністрування) [9].

Найпростішим підходом є ручний розподіл ключів – секретний ключ фізично передається від одного учасника іншому або розповсюджується через спеціально захищений канал зв’язку (наприклад, особисту зустріч чи кур’єрську доставку). Цей метод покладається на попередню довіру між сторонами та безпеку каналу передачі: якщо ключ потрапить до зловмисника на етапі обміну, то шифрування фактично втрачає сенс. Аналогічним є попередній обмін ключами – ключі заздалегідь генеруються і розподіляються уповноваженою особою або адміністратором до початку сеансу зв’язку між сторонами.

Для покращення масштабованості класичних схем було впроваджено симетричні протоколи розподілу ключів із використанням довіреного централізованого сервера. У таких протоколах кожен користувач попередньо встановлює спільний довгостроковий секретний ключ із центральним вузлом – так званим центром розподілу ключів (Key Distribution Center, KDC). Коли двом сторонам потрібно налагодити захищений зв’язок, KDC генерує для них новий спільний сеансовий ключ і передає необхідні дані для його використання (наприклад, зашифровані «квитки») [7]. Історично одним з перших подібних рішень став протокол Нідгема–Шрьодера (1978 р.), у якому для узгодження сеансових ключів залучався окремий сервер.

На основі цих ідей згодом створено низку систем автентифікації, зокрема відомий протокол Kerberos [7]. У таких схемах кількість необхідних ключів зменшується до лінійної залежності від числа користувачів (кожен зберігає лише один персональний ключ для зв’язку з KDC), проте виникає потреба повністю довіряти центральному серверу, який має бути надійно захищеним і постійно доступним. Більш того, процес встановлення спільного ключа вимагає проведення кількох раундів обміну повідомленнями за участю третьої сторони [9]. Це ускладнює протокол порівняно з прямим попереднім обміном ключами.

Одним із найвідоміших прикладів централізованої служби автентифікації на основі симетричних ключів є протокол Kerberos [7]. Його було розроблено у 1980-

х роках в Массачусетському технологічному інституті (MIT) в рамках проэкту Athena. Kerberos став відповіддю на потребу захищеної автентифікації у відкритих комп’ютерних мережах, де передавання паролів у незахищеному вигляді було неприпустиме [7]. У Kerberos кожен користувач і кожен мережевий сервіс реєструються на сервері автентифікації (KDC) і отримують спільний секретний ключ, відомий лише цьому серверу та відповідному суб’єкту [7]. Під час початкового входу (логіну) користувач вводить свій пароль, на основі якого KDC відновлює його довгостроковий ключ та здійснює перевірку автентичності. В результаті KDC видає користувачу спеціальний квиток-дозвіл – ticket-granting ticket (TGT), зашифрований ключем користувача. TGT слугує тимчасовим посвідченням особи і дозволяє користувачу надалі запитувати доступ до різних сервісів без повторного введення паролю.

Для доступу до конкретного сервісу користувач надсилає свій TGT (через службу видачі квитків – Ticket Granting Service) із запитом на доступ до цього сервісу. Якщо користувач успішно автентифікований, KDC видає квиток сервісу (service ticket) – дані, зашифровані секретним ключем запитуваного сервісу. Користувач пересилає цей квиток цільовому серверу, і сервер, розшифрувавши його своїм ключем, переконується в автентичності клієнта. Одночасно користувач отримує від сервера підтвердження його справжності (на основі наданого сеансового ключа).

Ця модель централізованої автентифікації лягла в основу багатьох сучасних рішень: протокол Kerberos (версія 5) став відкритим стандартом (RFC 4120) і був впроваджений у різних платформах та операційних системах (наприклад, в Windows Active Directory). Водночас схема Kerberos накладає певні вимоги та обмеження: необхідна синхронізація часу між усіма учасниками, а сам центр автентифікації є єдиною точкою відмови та потенційною мішенню для атак, що ускладнює адміністрування великомасштабних середовищ [7]. Поява криптографії з відкритим ключем докорінно змінила підходи до розповсюдження ключів. Замість

того щоб обмінюватися секретними симетричними ключами, сторони можуть використовувати відкриті ключі, які не потребують захисту при розповсюдженні. Головним завданням стало забезпечити автентичність таких відкритих ключів – тобто гарантувати, що отриманий ключ дійсно належить заявленому власнику.

Для вирішення цього завдання була створена інфраструктура відкритих ключів (PKI). PKI – це сукупність ролей, політик, апаратних і програмних засобів та процедур, необхідних для створення, управління, зберігання, розповсюдження та відкликання цифрових сертифікатів [8]. Узагальнену стуктуру PKI зображено на рисунку 1.1.

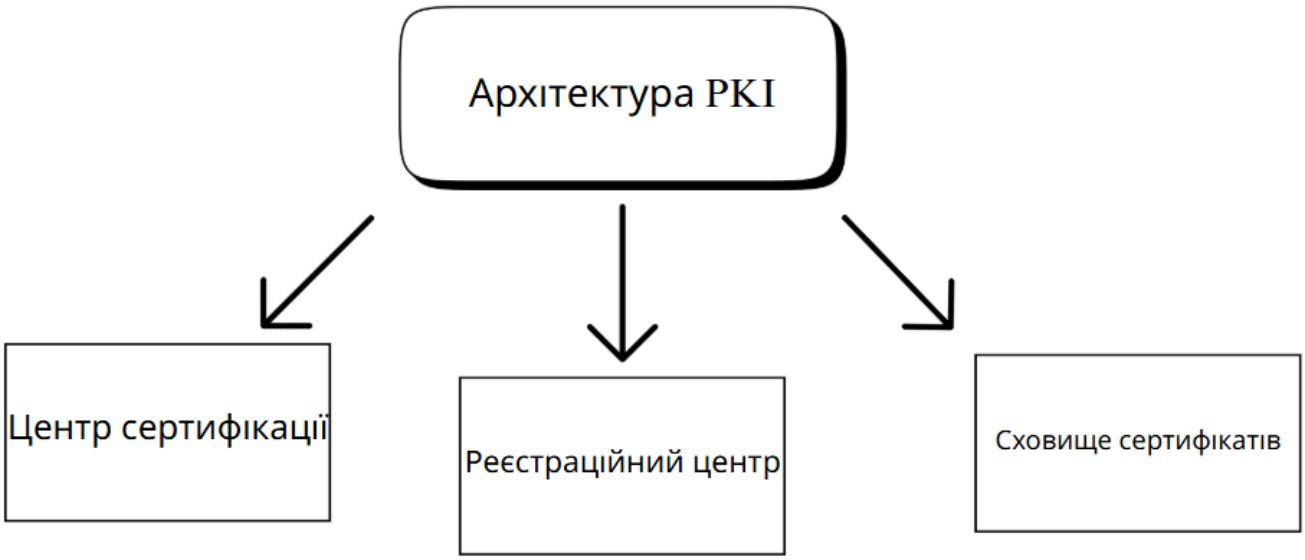

Рисунок 1.1– Структура PKI

В таблиці 1.1 наведено архітектуру PKI, а саме її основні компаненти та їх опис.

Таблиця 1.1 – Архітектура PKI

Компонент	**Опис**
Суб’єкт (користувач або сервер)	Сторона, яка використовує криптографічні ключі для шифрування, електронного підпису чи автентифікації.

Продовження таблиці 1.1

Центр сертифікації (CA)	Довірений вузол, що видає цифрові сертифікати, Верифікує особу суб'єкта та засвідчує зв'язок відкритого ключа з цією особою.
Реєстраційний центр (RA)	Допоміжна ланка, якій CA може делегувати перевірку документів та ідентифікацію суб'єктів перед видачею сертифіката.
Сховище сертифікатів	Репозиторій для зберігання та публікації виданих сертифікатів і списків відкликаних сертифікатів. Дозволяє перевіряти чинність та статус сертифікатів.

Ключовим елементом PKI є цифровий сертифікат – електронний документ, що зв'язує відкритий ключ з даними про власника (наприклад, ім'ям користувача чи назвою сервера) і завіряється цифровим підписом CA. Щоб довіряти такому сертифікату, сторона-отримувач повинна довіряти центру сертифікації, який його підписав.

Зазвичай використовується ієрархічна модель: існує кореневий CA, якому довіряють усі учасники системи, а він, у свою чергу, делегує повноваження підлеглим (проміжним) центрам сертифікації. Через ланцюжок підписів від кореневого до кінцевого сертифіката вибудовується ланцюжок довіри: довіряючи кореневому сертифікату, користувачі транзитивно довіряють усім сертифікатам, підписаним у межах цієї ієрархії [8]. PKI також підтримує механізми перехресної довіри, коли дві організації (або домени) взаємно визнають сертифікати один одного шляхом обміну довіреними кореневими сертифікатами. Інфраструктура відкритих ключів широко використовується для забезпечення безпеки у різних сферах. До типових застосувань PKI належать: протоколи TLS/SSL для захисту веб-транзакцій (HTTPS, інтернет-банкінг, електронна комерція), електронний цифровий підпис для підтвердження автентичності та цілісності електронних

документів і листів (технологія S/MIME), автентифікація користувачів у корпоративних мережах (наприклад, за допомогою смарт-карток чи апаратних токенів) та захищений обмін даними між організаціями.

1.4 Технологія блокчейн: принципи розподіленого реєстру та консенсусу

Блокчейн – це тип розподіленої цифрової бази даних (реєстру), що зберігає впорядкований ланцюжок записів (блоків), який постійно доповнюється новими блоками транзакцій без участі центрального посередника [11, 12]. Історично технологію блокчейн вперше було реалізовано у 2009 році як основу криптовалюти Bitcoin, запропонованої Сатоші Накамото в 2008 році [10]. Відтоді блокчейн еволюціонував у загальний підхід до ведення розподіленого реєстру – тобто спільного журналу транзакцій, яким керує децентралізована мережа вузлів без централізованого контролю [11]. У такому реєстрі кожен учасник мережі зберігає копію даних, а механізми консенсусу (див. далі) забезпечують узгодженість інформації між усіма вузлами.

В таблиці 1.2 зазначено ключові характеристики технології блокчейн.

Таблиця 1.2 Ключові характеристики блокчейн-технології

Характеристика	**Опис**
Децентралізація	Усі вузли мережі рівноправні, що усуває потребу в посереднику для підтвердження транзакцій.
Незмінність	Після підтвердження блоків дані неможливо змінити чи видалити.
Прозорість	Транзакції видимі усім учасникам мережі.

1.5 Структура та функціонування розподіленого реєстру

Розподілений реєстр у блокчейні представлений у вигляді безперервного ланцюга блоків, що зберігає всю історію транзакцій від моменту створення першого (генезис) блоку. Копія цього реєстру (блокчейну) зберігається на багатьох вузлах

мережі одночасно. Вузли (комп’ютери-учасники мережі) підключені за принципом peer-to-peer, тобто без централізованого сервера: обмін повідомленнями про транзакції і блоки відбувається безпосередньо між вузлами. Кожен вузол містить актуальну версію повного ланцюжка блоків або його частини та оновлює її в міру надходження нових даних згідно з правилами протоколу.

Алгоритм функціонування розподіленого реєстру такий: нові транзакції, створені користувачами, транслюються по всій мережі вузлів. Кожен вузол, отримавши транзакцію, перевіряє її коректність (наприклад, криптографічний підпис відправника, відсутність подвійного витрачання тощо).

Валідовані транзакції групуються у блоки. Спеціальні вузли (у відкритих блокчейнах їх називають «майнерами» або валідаторами) формують нові блоки за встановленими правилами і пропонують їх для додавання до реєстру. Блок містить набір транзакцій та службову інформацію і має посилання на попередній блок ланцюжка. Щойно новий блок згенеровано, він розсилається усім іншим вузлам мережі. Кожен вузол перевіряє отриманий блок: чи відповідає він протоколу (правилам формування блоків), чи валідні всі транзакції в блоці, чи правильний криптографічний геш попереднього блоку тощо. Якщо блок успішно проходить перевірки, вузол додає його до своєї копії блокчейну, продовжуючи тим самим ланцюжок. Завдяки цьому всі вузли поступово приходять до єдиної узгодженої версії реєстру – той самий порядок блоків і транзакцій у них.

У розподіленому середовищі можливі ситуації, коли різні вузли тимчасово мають різні версії ланцюжка (наприклад, якщо два блоки з’явилися майже одночасно в різних частинах мережі). Проте протокол визначає формальні правила досягнення згоди (консенсусу), які вузли використовують для вибору єдиної «правильної» версії ланцюга. Вузли, що тимчасово опинилися на відхиленій гілці, згодом переймають актуальний ланцюг і оновлюють свої копії. Такий підхід гарантує узгодженість даних без центрального координатора: консенсус

досягається за рахунок дотримання єдиного протоколу всіма чесними учасниками мережі.

2 ОГЛЯД ТА АНАЛІЗ ПІДХОДІВ ДО ВИКОРИСТАННЯ БЛОКЧЕЙНУ ДЛЯ ЗБЕРІГАННЯ І ПЕРЕДАЧІ КРИПТОГРАФІЧНИХ КЛЮЧІВ

2.1 Децентралізовані системи PKI на основі блокчейну

У традиційній моделі PKI використовується ієрархія центрів сертифікації (CA), які видають цифрові сертифікати, що пов’язують відкритий ключ з ідентичністю суб’єкта (наприклад, домену чи організації) [16]. Ця система ґрунтується на централізованій довірі: всі користувачі довіряють кореневим CA, а отже — і всім сертифікатам, підписаним ними. Клієнт (наприклад, браузер) перевіряє ланцюжок сертифікатів до довіреного кореня і вважає відкритий ключ дійсним, якщо підпис валідний. Для відкликання сертифікатів використовуються списки CRL або протокол OCSP.

Альтернативою є децентралізована модель Web of Trust, наприклад, PGP, де користувачі самостійно підписують ключі один одного. Така peer-to-peer система усуває єдину точку відмови, але має недоліки — складнощі з початковою перевіркою особи та відсутність уніфікованого механізму відкликання ключів. Тому вона не набула широкого поширення у веб-автентифікації.

Централізована PKI має вразливості: компрометація будь-якого CA загрожує цілісності всієї системи. Наприклад, у 2011 р. були зламані CA DigiNotar та Comodo, внаслідок чого видано сотні фальшивих сертифікатів [17]. Система CT (Certificate Transparency) лише реєструє видані сертифікати, але не запобігає їх несанкціонованому створенню і залежить від активності спільноти. Можливі атаки типу «розділеного світу», коли жертві показують фальшиву версію журналу.

Рішенням може стати децентралізація PKI за допомогою блокчейну — розподіленого реєстру, що забезпечує незмінність і прозорість даних завдяки консенсусу між вузлами. Така система усуває потребу в довіреній третій стороні: інформація про ключі зберігається в децентралізованій формі, захищеній

криптографічно. Блокчейн також виключає можливість split-world атак, оскільки всі учасники бачать одну й ту ж версію даних.У такій системі компрометація окремого вузла не порушить цілісності реєстру, а спроби несанкціонованої модифікації записів потребуватимуть контролю над більшістю мережі, що надзвичайно складно. Блокчейн також гарантує єдину глобальну версію даних: усі клієнти бачать однаковий реєстр, тож атаки типу «split-world» стають практично неможливими [17]. Таким чином, децентралізований підхід здатний забезпечити повну прозорість PKI: кожен виданий або відкликаний сертифікат фіксується в загальнодоступному реєстрі і може бути незалежно перевірений.

У блокчейн-орієнтованих системах перевірка сертифіката зводиться до пошуку запису в реєстрі та перевірки підпису транзакції, створеної власником домену. Це усуває потребу в довіреній третій стороні — достатньо довіряти самому блокчейну. На основі цієї ідеї розроблено моделі управління сертифікатами з прозорістю та відсутністю централізованого контролю.

CertCoin — децентралізована інфраструктура відкритих ключів, яка об’єднує відкритий реєстр і модель довіри без єдиної точки відмови. Використовує онлайн-ключ для TLS-з'єднань і офлайн-ключ — для оновлення/відкликання. У разі компрометації онлайн-ключа власник підписує транзакцію відкликання офлайн-ключем. Для пошуку відкликаних сертифікатів пропонуються криптографічні акумулятори, а для зберігання додаткової інформації — DHT.

Переваги: повна децентралізація, прозорість, стійкість до атак. Недоліки: потрібна адаптація клієнтів до Namecoin і вирішення проблеми первинної реєстрації доменів, інакше можливе кіберсквотерство.

CertLedger — гібридна модель, яка інтегрується з TLS. Всі операції PKI (видача, перевірка, відкликання) переносяться у блокчейн. Смарт-контракти перевіряють сертифікати на відповідність X.509 і підпис кореневим CA, фіксуючи їх як довірені. У разі відкликання публікується відповідна транзакція.

CertLedger створює глобальний реєстр сертифікатів і статусів, замінюючи CRL, OCSP і CT, забезпечуючи більшу прозорість та безпеку PKI.Для клієнтів (TLS-користувачів) інтеграція CertLedger виглядає наступним чином: під час встановлення з'єднання сервер домену може надати клієнту криптографічний доказ з блокчейну (наприклад, мерклеве підтвердження включення сертифіката в останній блок), який підтверджує, що сертифікат є зареєстрованим і не відкликаним [16]. Отримавши такий доказ, клієнт швидко його перевіряє (маючи доступ до заголовка недавнього блоку CertLedger, який можна отримати від мережі або з кешу) і впевнюється в актуальності сертифіката. Це усуває потребу у запитах до OCSP і власній перевірці шляху сертифікації – достатньо довіряти консенсусу блокчейну [16]. Згідно з оцінкою авторів, TLS-потиск руки з використанням CertLedger навіть перевершує за швидкістю стандартний, оскільки скорочує кількість перевірок і мережевих запитів [16]. Інша перевага – уніфікація довірчої бази: більше не потрібно зберігати локально сотні кореневих сертифікатів і періодично оновлювати їх список, адже актуальний перелік довірених центрів і їхній стан підтримується в самому блокчейні [16]. Це спрощує управління довірою: якщо якийсь CA скомпрометовано або порушено політики, більшість учасників мережі CertLedger (наприклад, представники різних браузерів, ОС, організацій) можуть консенсусом позначити його як недовірений, і зміна автоматично набуде чинності для всіх клієнтів.

Перевагу слід надавати алгоритмам консенсусу з швидкою фіналізацією блоків (наприклад, варіанти BFT або Proof-of-Authority), щоб уникнути ситуацій, коли клієнт перевірив статус сертифіката у непідтвердженому блоці, який пізніше було відкинуто [16]. Це може означати перехід до частково дозвільного блокчейну (permissioned blockchain) з відомими валідаторами, що потенційно зменшує децентралізацію. Ще одне питання – масштабованість: глобальний реєстр усіх TLS-сертифікатів у світі може нараховувати мільйони записів, і хоча зберігання гешів чи метаданих замість повного сертифіката економить місце, обсяг даних все одно

значний. Необхідно ефективно організувати зберігання і доступ, особливо для легких клієнтів [16].

Архітектура SCPKI передбачає, що кожен учасник (користувач, сервіс або домен) має пару ключів та унікальний ідентифікатор (можливо, прив'язаний до Ethereum-адреси або іншого ідентифікатора в системі). Смарт-контракт на блокчейні відповідає за збереження зв'язків довіри: коли один учасник «сертифікує» (підписує) відкритий ключ іншого учасника, відповідний запис додається до реєстру на блокчейні [3]. Це схоже на обмін довіреними підписами в PGP, але реалізовано прозоро і незмінно – всі підписані твердження про довіру зберігаються в смарт-контракті, доступні для перевірки будь-ким. У результаті формується граф довіри на блокчейні: вузли – це учасники та їхні ключі, а ребра – електронні підписи (сертифікати) від одних учасників на ключі інших. Для перевірки автентичності ключа певної особи (або сервера) потрібно знайти шлях у цьому графі від довіреного вами вузла до цільового вузла. Якщо такий шлях існує (наприклад, ваш друг підписав ключ адміністратора, а той – ключ цільового користувача), то ви можете довіряти цьому ключу транзитивно. Механізми довіри в SCPKI ґрунтуються на тому, що особу не підтверджує жоден централізований орган — натомість це роблять інші користувачі.

Блокчейн гарантує, що всі бачать єдину й актуальну картину довірчих зв'язків, а спроби приховати «зловмисний» ключ практично неможливі: якщо хтось створить фальшивий ключ на ім'я відомої особи, інші його не підпишуть, і він не матиме жодних шляхів довіри від чесних вузлів. Таким чином, SCPKI дозволяє виявляти «самозванців» (rogue keys) – будь-який ключ без підписів від відомих учасників одразу виглядає підозріло, а якщо раптом з'явиться підроблений підпис, це побачать всі і можуть відкликати довіру до компрометованого учасника [17]. Сховище ключів і сертифікатів у SCPKI організоване частково поза блокчейном: для збереження об'ємних даних (наприклад, самих сертифікатів чи додаткової інформації) пропонується використовувати розподілене сховище на кшталт DHT, а

в блокчейн записуються лише геші або індекси, що посилаються на ці дані [17]. Це економить ресурси і дозволяє масштабувати систему, зберігаючи при цьому незмінність прив’язок і довірчих заяв.

Переваги SCPKI: максимальна децентралізація і самоврядність учасників – користувачі самі контролюють свої ключі і довірчі зв’язки, не покладаючись на жодну центральну організацію [17]. Система прозора: всі можуть перевірити, хто кому довіряє, і виявити аномалії. Блокчейн слугує неупередженим арбітром, який автоматично виконує правила (смарт-контракт) і гарантує цілісність історії довіри.

Недоліки: модель Web of Trust, навіть підсилена блокчейном, стикається зі старими проблемами масштабування соціальної довіри. По-перше, складність ініціалізації довіри: новачку треба знайти достатньо вже довірених учасників, готових підписати його ключ, аби він став визнаним в системі. У глобальному масштабі це непросто, особливо для доменів чи організацій – можливо, доведеться передбачити механізми початкової верифікації поза системою (як-от підтвердження права на домен через DNS або інші канали). По-друге, розрахунок транзитивної довіри: визначити, чи достатньо довірених шляхів веде до ключа, може бути нетривіально (в PGP, наприклад, існують метрики «рівня довіри» та мінімальної кількості незалежних шляхів). Вартість – у публічних блокчейнах транзакції мають ціну, і хоча вона може бути незначною для окремої операції, сумарно витрати можуть стати значними, якщо, наприклад, кожні кілька місяців мільйони сертифікатів поновлюються і записуються в блокчейн. Інша проблема – сумісність з існуючою інфраструктурою. Щоб децентралізована PKI запрацювала в масштабах Інтернету, її мають підтримати виробники браузерів, ОС, сертифікаційні центри, стандартизуючі органи. На даний час традиційна PKI міцно вкорінена, і перехід вимагатиме поступового впровадження або співіснування систем.

Можливим є гібридний сценарій, де блокчейн виступає як додатковий рівень довіри — наприклад, у ролі добровільного журналу/аудитора поряд із CT. Повна відмова від центрів сертифікації та перехід до мережевої довіри вимагатиме

глибоких змін у протоколах і звичках користувачів. Також важливо враховувати ризики самої блокчейн-інфраструктури: атака 51%, вразливості консенсусу або баги в смарт-контрактах можуть серйозно загрожувати PKI. Тому зазвичай розглядають зрілі блокчейн-платформи або консорціумні реєстри з обмеженим колом валідаторів, що знижує рівень децентралізації, але підвищує керованість.

Щодо приватності: відкритий реєстр дозволяє фіксувати зловживання, але водночас розкриває зв'язки між суб'єктами. Для особистих чи корпоративних сертифікатів це може бути небажаним. Можливі варіанти – шифрування полів або нульові докази, проте це ускладнює систему та суперечить ідеї прозорості.

Існує безліч прототипів (CertCoin, CertLedger, SCPKI), але бракує узгодженого стандарту. Це природний етап для нової технології — у майбутньому можлива консолідація на базі найуспішніших рішень (наприклад, під егідою IETF). Блокчейн-PKI має потенціал розв'язати давні проблеми централізованих систем, проте наразі переважно перебуває на стадії експериментів і досліджень [15].

2.2 Протоколи обміну криптографічними ключами з використанням блокчейну

Класичний протокол Діффі–Геллмана (DH) дозволяє двом сторонам виробити спільний секретний ключ через відкрите мережеве з'єднання без попередніх спільних секретів. Такий ключ надалі використовується для шифрування подальшого обміну даними симетричними алгоритмами. Проте базова схема DH не забезпечує автентифікації сторін: протокол є «анонімним», тобто не підтверджує особу учасників обміну [18]. Це створює вразливість до атаки типу «людина посередині» (Man-in-the-Middle), коли зловмисник може перехопити публічні параметри обміну і підмінити їх, видаючи себе за кожну зі сторін. У відкритих мережах без додаткових заходів контролю цілісності та автентичності сторонній атакувальник здатен непомітно втрутитись у протокол обміну ключами та згенерувати власні спільні секрети з кожною жертвою, фактично компрометуючи

сеанс. Традиційний обмін ключами, зокрема DH, потребує додаткових механізмів захисту. Це, своєю чергою, вимагає інфраструктури відкритих ключів (PKI) або заздалегідь довірених центрів сертифікації, що у децентралізованих або відкритих середовищах може бути складно забезпечити. Інша проблема класичних схем узгодження ключів у відкритих мережах – необхідність синхронної взаємодії: обидва учасники повинні обмінятися повідомленнями безпосередньо. Якщо один з учасників тимчасово недоступний, встановити спільний секрет неможливо без посередників.

Блокчейн працює на основі децентралізованого консенсусу, що усуває потребу в центральному сервері чи авторитеті для перевірки ключів. Це підвищує стійкість системи до єдиної точки відмови та зловмисних дій з боку третьої сторони [3]. Однорангові учасники можуть довіряти даним у блокчейні, оскільки ті підтверджені більшістю вузлів мережі. Усі транзакції, записані в блокчейні, криптографічно зв’язані (через геш-ланцюжок) і підписуються особистим ключем відправника, як на рисунці 2.1.

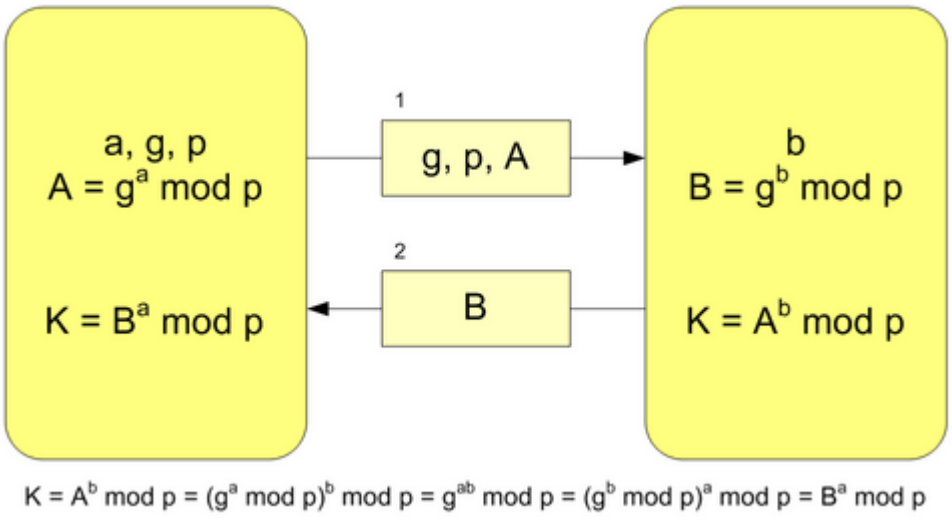

Рисунок 2.1 – Алгоритм Діффі – Геллмана

Це означає, що будь-які дані, передані в рамках протоколу обміну ключами, захищені від непомітної підміни. Блокчейн фактично забезпечує вбудовану

перевірку цілісності та автентичності повідомлень, ефективно запобігаючи атакам типу «людина посередині» [18]. На відміну від традиційного відкритого каналу, зловмисник не може непомітно змінити або підробити повідомлення: кожен учасник може перевірити достовірність отриманих через блокчейн даних за цифровим підписом відправника і записом транзакції.

Дані в блокчейні є відкритими (для відкритих блокчейнів) або доступними учасникам консорціуму (для особистих/дозволених блокчейнів). Алгоритм протоколу обміну ключами, якщо він реалізований як смарт-контракт, є відкритим для перевірки. Учасники можуть незалежно верифікувати правильність виконання протоколу, оскільки вся логіка та обмін повідомленнями фіксуються у реєстрі. Це сприяє побудові довіри між незнайомими сторонами: вони покладаються не один на одного, а на гарантії прозорості блокчейн-платформи [18].

Блокчейн виступає постійним сховищем повідомлень, завдяки чому обмін ключами може відбуватися асинхронно. Сторона А може опублікувати свої дані для обміну (наприклад, публічний параметр) в блокчейні, і ці дані будуть там збережені, поки сторона В не прочитає їх і не відповість своєю частиною протоколу. Таким чином, учасники не зобов’язані бути одночасно online для встановлення спільного ключа – достатньо доступу до блокчейну [18]. Крім того, надійність розподіленої мережі забезпечує доступність повідомлень: навіть у разі збою окремих вузлів або спроб цензурувати трафік, дані в блокчейні, як правило, залишаються доступними завдяки дублюванню на багатьох вузлах.

Всі кроки обміну ключами, які виконуються через транзакції, автоматично протоколюються. Це означає, що пізніше можна перевірити, коли і які відкриті ключі обмінювались, чи відбулася успішна установка спільного секрету, тощо. Невідмовність гарантується тим, що запис у блокчейні неможливо редагувати або видалити заднім числом: жоден учасник не може заперечити факт відправлення свого публічного компонента обміну, якщо транзакція підтверджена в ланцюгу

[19]. Такий аудит дозволяє виявити або відстежити спроби несанкціонованих дій, а також слугує доказовою базою в разі суперечок.

Перелічені властивості роблять блокчейн привабливим середовищем для реалізації протоколів узгодження ключів у розподілених системах. Нижче розглядаються кілька підходів і протоколів, що використовують блокчейн для обміну криптографічними ключами, та їх особливості.

У SmartDHX вирішено проблему генерації випадкових чисел у блокчейні: замість небезпечної генерації у смарт-контракті приватний параметр генерується локально на клієнті, а до блокчейну передається лише відкритий ключ. Смарт-контракт організовує обмін цими ключами між сторонами (A і B), дозволяючи кожній з них локально обчислити спільний секрет без зберігання його в блокчейні [1, 17].

Блокчейн виступає надійним транспортом, що гарантує цілісність та доставку публічних параметрів. Протокол SmartDHX підтримує асинхронний обмін — учасники можуть не бути онлайн одночасно, оскільки повідомлення зберігаються у блокчейні до прочитання. Також забезпечується захист від атак: усі дані підписані та публічно доступні, що унеможливлює підміну ключів [18]. Блокчейн надає гарантії цілісності й автентичності, усуваючи ризик непомітної атаки посередника – зловмисник не може змінити переданий через смарт-контракт параметр, не порушивши підпис чи консенсус, тому протокол вважається стійким до MitM-атак у такому середовищі [17]. Важливою особливістю SmartDHX є підтримка багатостороннього обміну. Авторами узагальнено двосторонній протокол на випадок n учасників [17]. Для цього було реалізовано координаційну логіку у смарт-контракті, що дозволяє групі з кількох вузлів спільно виробити один секретний ключ. Кожен учасник публікує свій компонент, а контракт обчислює або забезпечує передачу необхідних проміжних значень таким чином, що всі учасники врешті отримують спільний секрет. Це демонструє гнучкість підходу: на блокчейні

можна реалізувати навіть складні криптографічні протоколи з багатьма сторонами [17].

Прототип SmartDHX було реалізовано мовою Solidity та розгорнуто в тестовій мережі Ethereum для оцінки ефективності [17]. Результати показали коректність роботи: кілька учасників успішно обмінялися ключем, жодних даних про секрет не було записано у блокчейн (лише проміжні публічні значення). При цьому виявлено неминучий компроміс: ціна децентралізації – зниження продуктивності. Виконання крипто-операцій і зберігання повідомлень в середовищі блокчейну потребує більше часу, ніж традиційний обмін по мережі, через накладні витрати на транзакції та обробку блоків [17]. Зокрема, узгодження ключа в SmartDHX займає додатковий час на підтвердження транзакцій майнерами, а також супроводжується витратами газу (комісії) за виконання смарт-контракту. Для групового обміну (кілька учасників) вдалося зменшити кількість необхідних транзакцій (порівняно з попарним обміном між всіма учасниками), однак зросло навантаження обчислень у самому контракті, що теж потребує ресурсів [1]. Попри ці накладні витрати, SmartDHX є важливим доказом концепції: вся логіка протоколу DH може бути перенесена на блокчейн без втрати безпеки. Це відкриває шлях до нових DApp, у яких вузли можуть динамічно встановлювати спільні секрети повністю через блокчейн, шифруючи подальший обмін даними без участі зовнішніх довірених сервісів.

2.3 Прихований обмін ключами: протокол BCDH

Блокчейн може використовуватися не лише для підвищення безпеки традиційних обмінів, а й для реалізації прихованих каналів зв’язку. У 2024 році запропоновано схему BCDH (Blockchain-Based Covert Diffie–Hellman) для прихованого узгодження ключів на основі блокчейну [18]. Актуальність цього підходу пов’язана з потребами прихованого (ковертного) зв’язку: відправник і отримувач хочуть обмінятися секретним ключем так, щоб сторонній спостерігач

навіть не зміг визначити самого факту встановлення секретного каналу. Звичайні методи обміну (навіть захищені) непридатні для такої задачі, оскільки характерний трафік протоколу Діффі–Геллмана або обмін сертифікатами явно сигналізує про криптографічну взаємодію. Наприклад, аналіз мережевого трафіку може виявити типові повідомлення узгодження ключа, що розкриє наявність прихованого зв'язку між сторонами. BCDH розв'язує цю проблему, дозволяючи двом сторонам негласно домовитися про ключ через блокчейн так, що спостерігач не помітить очевидного обміну ключовими даними [18]. Основна ідея BCDH полягає в тому, що відправник і одержувач не взаємодіють напряму, а координують свої дії через транзакції в блокчейн-мережі.

Після початкового налаштування сторони можуть обмінюватися кількома секретними ключами через блокчейн без явного «рукостискання» протоколу DH [18]. Реалізація базується на ECDH та цифрових підписах: відкриті ключі вбудовуються в звичайні транзакції так, що для сторонніх вони виглядають як звичайні фінансові операції. Лише учасники, що знають схему, можуть витягнути потрібні дані й обчислити спільний секрет. Це дозволяє обмінюватися ключами без прямого зв'язку, спостерігаючи за транзакціями в блокчейні.

Протокол BCDH підвищує прихованість, використовуючи кілька блокчейнів: етапи обміну розподіляються між ними, ускладнюючи відстеження й підвищуючи надійність. Якщо одна мережа недоступна, процес триває в інших.

Автори оцінили схему за ступенем прихованості, криптографічною безпекою, стійкістю до модифікацій і ефективністю. Прототип на Ethereum показав, що система придатна до практичного використання і забезпечує високий рівень конфіденційності [2, 18].

2.4 Застосування в IoT: контроль доступу до даних та узгодження ключів

Інтернет речей (IoT) – це середовище, де проблема безпечного обміну ключами та управління доступом до даних стоїть особливо гостро. IoT-системи

зазвичай складаються з великої кількості пристроїв, що генерують і передають дані через відкриті мережі, часто без можливості повністю довіряти інфраструктурі. Блокчейн знаходить застосування в IoT як засіб децентралізованого контролю доступу, розподілу ключів і протоколювання дій, усуваючи залежність від центрального серверу чи хмари. Ключові компоненти рішення [18]:

Перед передаванням даних датчик (або шлюз, що агрегує дані від групи пристроїв) і хмарний сервер проходять взаємне підтвердження особи, після чого виконують криптографічний протокол узгодження спільного сеансового ключа. Реалізація базується на методах з використанням білінійних пар та складної криптографічної задачі – дискретного логарифма на білінійних групах (DBDH) [3]. Протокол гарантує, що і пристрій, і сервер поділяють секретний сеансовий ключ, відомий лише їм, причому сторонній спостерігач не може його обчислити. Наявність взаємної автентифікації означає, що обидві сторони переконуються в законності одна одної (ні «підроблений» датчик, ні фальшивий сервер не зможуть пройти перевірку), що особливо важливо в IoT, де пристрої можуть бути захоплені або підмінені. Узгоджений сесійний ключ використовується для захисту каналу передачі даних (наприклад, для шифрування самих даних перед завантаженням).

Після встановлення сеансового ключа IoT-шлюз шифрує зібрані дані цим ключем і передає їх до хмарного серверу для зберігання [3]. Додатково застосовується атрибутно-орієнтоване шифрування з політикою шифрування (Ciphertext-Policy ABE): кожен фрагмент даних зашифровано таким чином, що розшифрувати його зможе лише той користувач, чий набір атрибутів (роль, права доступу тощо) відповідає політиці, зазначеній при шифруванні [3]. Блокчейн у цій схемі використовується для управління атрибутами і політиками доступу, а також для усунення довіри до єдиного сервера при зберіганні – розподілене зберігання і підтвердження прав доступу через смарт-контракт виключає можливість змови або помилки адміністратора, яка б надала доступ неавторизованим особам [3].

Коли шлюз надсилає зашифровані дані до хмари, він формує цифровий підпис (наприклад, геш від даних з міткою часу), який публікує в блокчейні [3]. Цей запис слугує верифікаційним слідом: будь-який споживач даних надалі може звернутися до блокчейну, щоб перевірити, що дані не були модифіковані (геш співпадає) і коли саме вони були завантажені. Крім того, до реєстру може заноситися і службова інформація, наприклад ідентифікатор атрибутної політики, щоб користувачі знали, які умови потрібні для доступу. Завдяки аудиту блокчейном забезпечуються невідмовність (відправник не може заперечити факт надсилання даних, оскільки його підпис збережено) та відповідальність: дії всіх сторін зафіксовані, що підвищує довіру до системи.

Побудована таким чином система була піддана як формальному, так і неформальному аналізу безпеки [3]. Формальна перевірка (з використанням інструменту AVISPA та моделювання атак на стійкість шифру) підтвердила, що протокол витримує широкий спектр атак, характерних для IoT-середовища – включно з атаками на вгадування ключів або паролів і спробами відстеження дій учасників (трасування) [3]. Неформальний аналіз також не виявив слабких місць у сценаріях, таких як підміна пристроїв, повторне відтворення повідомлень чи компрометація окремих вузлів. За результатами оцінки, запропонований протокол безпечніший щодо атак типу вгадування та відстеження, ніж раніше відомі аналоги, і водночас забезпечує всі необхідні функції – взаємну автентифікацію учасників та узгодження сесійного ключа для захисту каналу [3]. Це демонструє універсальність підходу: блокчейн може слугувати інфраструктурою для безпечного узгодження ключів і управління доступом у будь-яких розподілених середовищах, від фінансових застосунків до промислового IoT.

2.5 Переваги та недоліки застосування блокчейну для управління ключами

Блокчейн-технологія все частіше розглядається як альтернативний підхід до управління криптографічними ключами та інфраструктури відкритих ключів (PKI).

Традиційна PKI покладається на довірені центри сертифікації, що створює єдиний центр довіри та потенційну точку відмови. Натомість блокчейн пропонує децентралізований реєстр для зберігання й управління ключами та сертифікатами, що унеможливлює несанкціоновану зміну даних завдяки криптографічному захисту та консенсусу мережі. У цьому розділі розглядаються основні переваги використання блокчейну для управління ключами (цілісність, децентралізація, прозорість аудиту, незмінність), різні архітектурні підходи (on-chain vs. off-chain зберігання даних) та їх технічні компроміси, а також ключові проблеми безпеки і продуктивності (масштабованість, затримки, приватність). Окремо проаналізовано практичні обмеження і типові загрози при впровадженні блокчейну в систему управління ключами (перевантаження мережі, централізація консенсусу, фрагментація імен). На основі цього робляться висновки щодо доцільності застосування блокчейну для управління ключами в різних сценаріях.

Зберігання криптографічних ключів або сертифікатів у розподіленому реєстрі гарантує їхню цілісність: записи, підтверджені консенсусом блокчейну, не можуть бути непомітно змінені чи видалені сторонніми особами. Блокчейн виступає як незмінний журнал: одного разу додана інформація (наприклад, відкритий ключ чи сертифікат) стає постійною частиною ланцюга блоків. Це унеможливлює несанкціоноване корегування ключів або сертифікатів – будь-яка спроба фальсифікувати дані вимагатиме контролю над більшістю вузлів мережі, що практично нездійсненно в відкритих блокчейнах. Таким чином, забезпечується висока довіра до цілісності записів ключів: користувачі можуть бути впевнені, що отриманий з блокчейну відкритий ключ не було підмінено чи скомпрометовано під час зберігання. Крім того, криптографічний зв’язок записів (геш-ланцюг) забезпечує перевірку цілісності всього журналу від першого до останнього блока.

Блокчейн усуває потребу в центральному авторитеті (такому як центр сертифікації) для підтвердження достовірності ключів. Реєстр ключів ведеться колективно множиною незалежних вузлів, тому немає єдиного центру, вихід з ладу

або компрометація якого може поставити під загрозу всю систему. Це елімінація єдиної точки відмови, що особливо цінно в PKI: якщо в традиційній моделі злам або помилка CA може призвести до масових атак (як це траплялося у випадках DigiNotar, DigiCert Sdn. Bhd. тощо), то в блокчейн-ПКІ зловмиснику довелося б скомпрометувати більшість учасників мережі, що набагато складніше. Децентралізована природа системи також ускладнює цензуру або маніпуляцію: жоден вузол не володіє монопольним контролем над записами ключів. Завдяки цьому підвищується надійність і відмовостійкість інфраструктури управління ключами.

Оскільки блокчейн зберігає повну історію всіх змін, можна легко встановити, хто, коли і яку дію виконав з ключем. Кожен запис підписаний криптографічно (через механізм ключів учасників мережі), що забезпечує автентичність джерела даних. У контексті управління ключами це означає, що будь-яка операція (наприклад, відкликання компрометованого ключа або передача прав адміністрування) матиме незаперечний слід в журналі. Це спрощує розслідування інцидентів безпеки і сприяє дотриманню вимог регуляторів щодо обліку дій з ключами. Трасованість також полегшує виявлення і запобігання атак: наприклад, спроба повторно зареєструвати вже відкликаний ключ чи сертифікат буде помітною в історії транзакцій.

Традиційна система сертифікатів стикається з проблемою затримок у відкликанні (розповсюдження списків відкликаних сертифікатів, перевірка через OCSP тощо). У блокчейн-рішенні відкликання сертифіката або ключа може бути реалізоване шляхом додавання відповідного запису до реєстру, що миттєво стає видимим для всіх вузлів та клієнтів. Неможливість видалити чи приховати цей запис гарантує, що жоден зловмисний або недобросовісний центр не зможе приховати факт компрометації ключа. Батьківський вузол (наприклад, в ролі центру сертифікації) у блокчейн-PKI має змогу помітити сертифікат як відкликаний; якщо він цього не зробить, сам факт бездіяльності буде публічно зафіксовано. Таким

чином, блокчейн спрощує реалізацію надійного механізму відкликання та підвищує загальний рівень довіри до актуальності інформації про статус ключів.

Один з підходів до управління ключами за допомогою блокчейну – це повне ончейн-зберігання даних, коли відкриті ключі, сертифікати або їхні геші зберігаються безпосередньо у блокчейні. Перевагою є те, що всі властивості блокчейну (незмінність, цілісність, доступність для перевірки) повною мірою поширюються на ці дані. Ончейн-записи одразу захищені механізмами консенсусу: наприклад, запис про новий ключ буде IMMUTABLE (незмінним) і загальнодоступним для верифікації всіма учасниками мережі. Це виключає можливість атаки типу “маніпуляція каталогом ключів” або приховування компрометації ключа. Отже, ончейн зберігання забезпечує максимум безпеки та прозорості, але за рахунок швидкодії, масштабованості та вартості.

Альтернативний підхід – зберігати ключові дані поза блокчейном, залишаючи в самому блокчейні лише посилання або криптографічні підтвердження (наприклад, геші) цих даних. Реалізація може бути такою: відкриті ключі та сертифікати зберігаються у зовнішньому репозиторії або базі даних (централізованій чи розподіленій, наприклад, у файловій мережі IPFS), а в блокчейні фіксується лише їх криптографічний геш або індекс. Перевагою цього підходу є значне зменшення навантаження на блокчейн: обсяг даних та кількість транзакцій скорочуються, що підвищує швидкодію та масштабованість системи. Витрати на транзакції також нижчі, оскільки замість повних даних записуються компактні геші. Крім того, конфіденційні дані (наприклад, деталі сертифіката або особисту інформацію) можна тримати поза відкритим реєстром, зберігаючи особистість. Цей підхід має і недоліки, головний з яких – необхідність довіри до зовнішнього сховища. Дані поза блокчейном не захищені автоматично механізмами консенсусу, їх цілісність і доступність повинні забезпечуватися додатковими заходами. Геш на блокчейні гарантує, що підробку можна виявити, але не запобігає самій спробі підміни чи видалення даних у зовнішньому сховищі.

3. ПРОЄКТУВАННЯ СИСТЕМИ ДЛЯ РОЗПОДІЛЕНОГО ЗБЕРІГАННЯ ТА ПЕРЕДАЧІ КЛЮЧІВ ІЗ ВИКОРИСТАННЯМ БЛОКЧЕЙН

3.1 Вимоги до системи та загальна концепція

Система управління криптографічними ключами призначена для забезпечення надійного життєвого циклу криптографічних ключів в організації. Основними цілями такої системи є гарантування безпеки ключів (конфіденційності їх значень та цілісності), доступності потрібних ключів для авторизованих користувачів і процесів, а також належного контролю їх використання. СКК повинна підтримувати потрібний рівень криптографічного захисту, що відповідає вимогам конкретних застосувань і політикам безпеки організації, при цьому не створюючи надмірного навантаження чи перешкод у роботі користувачів. Іншими словами, система має забезпечувати баланс між високим рівнем захищеності ключів, продуктивністю та зручністю експлуатації.

Відповідно до рекомендацій NIST SP 800-130, безпечна система управління ключами повинна підтримувати ключові функції життєвого циклу: генерацію ключів, їх захищене зберігання, розповсюдження (дистрибуцію) визначеним отримувачам та надійне знищення після закінчення строку служби. Для кожної з цих функцій встановлено низку вимог і контролів. Нижче наведено основні функціональні вимоги до СКК за цими напрямами:

Генерація ключів: СКК повинна генерувати криптографічні ключі з використанням криптостійких методів, дотримуючись затверджених стандартів для відповідних алгоритмів. Зокрема, генерація ключових пар та симетричних ключів має здійснюватися на основі надійних генераторів випадкових чисел, призначених для криптографії.

Зберігання ключів: Система має безпечно зберігати секретні та особисті ключі протягом усього періоду їх використання. Для цього застосовуються

криптографічні модулі та сховища, що забезпечують ізоляцію ключового матеріалу від несанкціонованого доступу. Згідно з NIST SP 800-130, СКК повинна використовувати перевірені криптографічні модулі (апаратні чи програмні) для генерації, зберігання та використання ключів. Ключі у сховищі мають зберігатися в шифрованому вигляді або з використанням механізмів контролю доступу, щоб виключити можливість розголошення ключа у відкритому вигляді.

Розповсюдження (дистрибуція) ключів: СКК повинна забезпечувати безпечну передачу криптографічних ключів від місця їх створення або зберігання до авторизованих отримувачів. Як зазначено в NIST SP 800-130, система, ймовірно, буде задіяна в мережевому оточенні для розподілу ключів кінцевим користувачам або пристроям, тому необхідно застосувати відповідні мережеві засоби безпеки для захисту каналів передачі. Перед експортом ключа із захищеного модуля система повинна застосувати криптографічні заходи захисту – такі як шифрування ключа іншими ключами (key wrapping), цифрове підписування або використання протоколів розподілу ключів – щоб гарантувати конфіденційність і цілісність ключового матеріалу під час транспортування. Таким чином, лише належним чином уповноважені суб’єкти можуть отримати ключ, і його передача не компрометує безпеку.

У сучасних системах безпеки СКК часто функціонує у взаємодії з інфраструктурою відкритих ключів (Public Key Infrastructure, PKI), зокрема використовуючи сертифікати X.509 для асоціювання відкритих ключів з суб’єктами (користувачами, серверами тощо). RFC 5280 визначає стандартизовані формати X.509 сертифікатів відкритих ключів та відповідних списків відкликаних сертифікатів (CRL), а також правила обробки сертифікаційних ланцюжків і статусу відкликання. Таким чином, вимоги до СКК щодо інтеграції з X.509 включають підтримку цих стандартів і процесів PKI.

По-перше, система повинна забезпечувати можливість генерації ключових пар для подальшої сертифікації. СКК має або виступати в ролі центру сертифікації

(Certificate Authority, CA), або тісно взаємодіяти з зовнішнім CA, щоб випускати X.509-сертифікати для згенерованих ключів. Це означає підтримку формування запитів на сертифікат (CSR) та збереження виданих сертифікатів разом із відповідними закритими ключами. Формат сертифікатів і полів у них має відповідати профілю RFC 5280, зокрема включати коректні розширення (наприклад, Key Usage, Extended Key Usage, політики сертифікатів тощо) залежно від призначення ключа.

По-друге, СКК має забезпечувати високий рівень захисту особистих ключів, особливо ключів центрів сертифікації. Компрометація закритого ключа CA є однією з найнебезпечніших подій для інфраструктури відкритих ключів, оскільки зловмисник у такому разі може видавати підроблені сертифікати чи списки відкликаних сертифікатів від імені довіреного центру. Тому RFC 5280 наголошує на застосуванні сильних технічних засобів (наприклад, апаратних криптомодулів, захищених від несанкціонованого доступу) та управлінських процедур (розмежування обов'язків адміністраторів, резервне копіювання ключів тощо) для охорони особистих ключів CA. Існують дві принципово різні архітектурні моделі систем управління ключами: централізовані та децентралізовані. У централізованому підході всі основні функції СКК виконує єдиний сервер або кластер під єдиним управлінням – тобто існує центральне сховище ключів і сервіс, через який здійснюється генерування, зберігання та розподіл ключового матеріалу. Натомість децентралізований підхід передбачає розподіл функцій управління ключами між багатьма вузлами або учасниками мережі, без одного довіреного центру. Кожен із цих підходів має свої переваги і недоліки, і вибір залежить від вимог конкретної системи.

Більшість існуючих промислових та корпоративних рішень з управління ключами є централізованими. Як зазначають Kuzminykh та співавт. (2021), сучасні інструменти управління ключами (такі як HashiCorp Vault, Square Keywhiz, OpenStack Barbican, Pinterest Knox, CyberArk Conjur тощо) реалізовані у вигляді

централізованих сховищ ключів з API-доступом і включають мінімально необхідний набір функцій: наявність захищеної бази даних для зберігання ключів, модель автентифікації та авторизації для управління доступом до ключів, ефективне логування операцій з ключами та програмний інтерфейс (API) для інтеграції зі сторонніми застосунками. З огляду на наведені вимоги та характеристики, розглянемо, якою мірою блокчейн-орієнтована концепція СКК задовольняє поставлені вимоги, а в чому, навпаки, створює додаткові труднощі у порівнянні з традиційними підходами.

Вимоги, які задовольняються блокчейн-підходом. Використання блокчейну в системі управління ключами безпосередньо адресує проблему надійності та відмовостійкості. За рахунок розподілу даних між багатьма вузлами, блокчейн-СКК не має єдиного центру, від якого залежить вся система, – це підвищує доступність сервісу управління ключами. Навіть якщо окремі вузли виходять з ладу або піддаються атаці, інші вузли продовжать обслуговувати запити, зберігаючи цілісність системи. Такий підхід відповідає вимозі безперервності роботи СКК у критичних застосуваннях. Крім того, безпека розповсюдження ключів значно посилюється: сама природа блокчейну (розподілений реєстр) забезпечує захищену передачу даних між учасниками мережі. Записи про операції з ключами (наприклад, видачу нового ключа або відкликання існуючого) можуть зберігатися у формі транзакцій блокчейну. Завдяки криптографічним механізмам побудови блокчейну, ці транзакції є незмінними та пов’язаними з попередніми (через геш-ланцюжок блоків), що гарантує цілісність історії операцій. Це фактично виконує функцію аудиту: будь-які зміни або спроби несанкціонованих дій з ключами стануть помітними всім учасникам. Таким чином, вимога прозорого логування та моніторингу доступу до ключів виконується автоматично – блокчейн виступає незмінним журналом всіх ключових подій.

Розподілена довіра – ще один аспект, де блокчейн-концепція відповідає цілям безпеки. Не потрібно повністю покладатися на одну довірену сторону

(адміністратора або сервер); натомість довіра децентралізована між багатьма учасниками. Це зменшує ризик внутрішньої зловмисності або помилки, оскільки жоден вузол не може одноосібно згенерувати або змінити ключові дані без відома інших. Такий підхід особливо корисний в міжорганізаційних взаємодіях або у відкритих мережах, де учасники не мають спільного адміністратора. В контексті вимог інтеграції з PKI, блокчейн може слугувати альтернативним або додатковим механізмом поширення сертифікатів та статусів відкликання.

Попри наведені переваги, впровадження блокчейн у СКК створює певні труднощі у забезпеченні окремих вимог. Перш за все, вимога захищеного зберігання ключів стає більш складною: блокчейн за своєю суттю є відкритим журналом, тому не може використовуватися для зберігання самих секретних ключів у відкритому вигляді. Фактичне зберігання особистих ключів все одно доведеться організувати поза блокчейном (наприклад, на клієнтських пристроях або у захищених сховищах, які інтегровані з блокчейном). Це додає рівень складності, оскільки система стає гібридною: блокчейн відповідає за реєстрацію операцій і розподіл відкритих даних (наприклад, відкритих ключів або гешів ключів), а окремі компоненти – за локальне зберігання секретів. Необхідно гарантувати, що ці компоненти належним чином захищені (наприклад, шифруванням, HSM, тощо) і синхронізовані з блокчейн-реєстром.

Продуктивність та масштабування також можуть стати викликом. Хоча блокчейн добре масштабується географічно та за кількістю учасників, швидкодія окремих операцій з ключами буде нижчою, ніж у централізованих системах, через накладні витрати на досягнення консенсусу. Наприклад, видача нового ключа або сертифіката в блокчейн-СКК може вимагати підтвердження декількома вузлами і включення транзакції в блок, що займає час (залежно від обраного алгоритму консенсусу і параметрів мережі). Для деяких застосувань (реального часу, з високою частотою транзакцій) це може стати проблемою, якщо блокчейн не забезпечує достатньої пропускної здатності.

Ще одна складність – безпечне видалення ключів (знищення) згідно з вимогами. У традиційному СКК видалення секретного ключа означає знищення усіх його копій у сховищах. У блокчейн же дані, одного разу записані, залишаються назавжди (принаймні в межах типового блокчейну, що гарантує незмінність). Якщо в розподіленому реєстрі зберігається якась інформація, пов'язана з ключем (навіть геш чи зашифрований ключ), повністю видалити її неможливо – можна лише позначити ключ як відкликаний або більше не використовуваний.

Інтеграція з існуючим PKI – ще один аспект, де блокчейн-підхід може ускладнити виконання вимог. Як зазначалося, X.509 інфраструктура покладається на ієрархічну модель довіри. Вбудувати децентралізований механізм в цю модель – нетривіальна задача. Наприклад, аби браузери чи інші клієнти довіряли сертифікату, емітованому через блокчейн-консенсус, необхідно або щоб такий сертифікат був підписаний звичним кореневим центром сертифікації, або щоб клієнти були модифіковані для підтримки нового механізму перевірки. На практиці це означає, що блокчейн-СКК, скоріш за все, доведеться використовувати разом із традиційними центрами сертифікації (або принаймні мати один довірений кореневий ключ у мережі, відомий всім учасникам). Така гібридність може породжувати додаткові точки відмови чи вразливості, якщо реалізована неправильно.

3.2 Архітектура запропонованої системи

Запропонована система являє собою програмний прототип, реалізований мовою Python, що імітує роботу блокчейн-системи для обміну криптографічними ключами між двома вузлами. Архітектуру запропонованої системи наведено на рисунку 3.1. Основні компоненти системи перелічені в таблиці 3.1.

Основна мета цієї системи полягає в забезпеченні захищеного обміну даними (зокрема, секретними повідомленнями) між двома учасниками без потреби в централізованому довіреному посереднику. Концепція полягає у використанні

технології розподіленого реєстру (блокчейну) для зберігання відкритих криптографічних ключів учасників і зашифрованих повідомлень, що обмінюються між ними. Такий підхід дозволяє досягти ключових властивостей безпеки – незмінності записів, автентичності джерела повідомлень та конфіденційності переданої інформації.

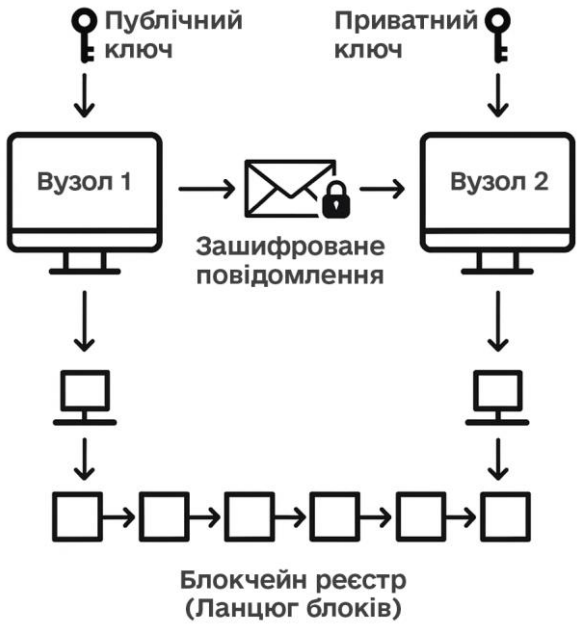

Рисунок 3.1 – Архітектура запропонованої системи

Система складається з двох рівноправних вузлів (учасників), кожен з яких має власну пару криптографічних ключів (відкритий та особистий ключ) і підтримує локальну копію спільного ланцюга блоків. Взаємодія між вузлами здійснюється шляхом обміну блоками через мережеве з’єднання або імітований канал зв’язку.

Кожен вузол може формувати нові блоки з транзакціями (наприклад, публікацією свого відкритого ключа чи зашифрованим повідомленням) та передавати їх іншому вузлу. Обидва вузли перевіряють отримані блоки на коректність та додержання правил протоколу, після чого додають їх до свого ланцюга. Таким чином, обидва учасники завжди підтримують синхронізовану,

ідентичну копію блокчейну, що служить єдиним достовірним джерелом даних про обмін ключами і повідомленнями.

Таблиця 3.1 – Компоненти системи

Компонент	**Призначення**	**Мова та бібліотека**
Blockchain	Структура з геш-зв’язком між блоками	Python, hashlib, json
CryptoModule	Генерація ключів, підпис, шифрування/дешифрування	Python, cryptography
Node	Мережева логіка, обробка блоків і перевірка транзакцій	Python, socket, os

Для реалізації функціональності система використовує кілька пов’язаних структур даних, що забезпечують роботу блокчейну:

1) Блок є базовою одиницею даних у ланцюжку. Він містить службову та корисну інформацію: заголовок і список транзакцій. До заголовку блоку входять принаймні ідентифікатор або порядковий номер блоку, мітка часу його створення та криптографічний геш попереднього блоку (для генезис-блоку попередній геш може бути нульовим або заздалегідь визначеним). Також блок містить власний геш, обчислений на основі вмісту блоку (заголовка і транзакцій) – цей геш використовується іншими блоками для зв’язування в ланцюжок. У прототипі може застосовуватися спрощена схема, де кожен блок містить одну транзакцію, що полегшує перевірку та синхронізацію.
2) Блокчейн – це впорядкована послідовність блоків, пов’язаних між собою через криптографічні геші. Кожен наступний блок містить геш попереднього, тим самим утворюється ланцюжок, який неможливо змінити без порушення цілісності. Усі вузли мережі зберігають актуальну копію цього ланцюга. Додавання нового блоку здійснюється тільки в кінець ланцюжка (на основі

останнього блоку), що забезпечує хронологічний запис подій. Блокчейн виступає журналом транзакцій системи, в якому можна простежити всі кроки обміну ключами та повідомленнями від початку функціонування системи.

3) Транзакція – це структурована одиниця даних, яка відображає окрему дію або подію в системі. У рамках запропонованої системи можна виділити два основних типи транзакцій: (1) транзакція оголошення відкритого ключа і (2) транзакція передачі повідомлення. Транзакція містить поля, необхідні для інтерпретації цієї дії: тип транзакції, дані (значення відкритого ключа або зашифроване повідомлення), а також ідентифікатор відправника (яким може слугувати відкритий ключ або інший унікальний ID вузла). Кожна транзакція, що формується вузлом, підписується його особистим ключем, що дозволяє будь-якому отримувачу перевірити цю транзакцію на автентичність (переконатися, що її створив саме заявлений вузол і дані не були змінені).
4) Повідомлення у контексті транзакцій повідомлення – це корисні дані, що передаються між вузлами. Перед додаванням до транзакції текст повідомлення шифрується відкритим ключем отримувача. У транзакції може зберігатися як сам зашифрований текст, так і додаткова службова інформація (наприклад, ознака чи ідентифікатор одержувача, якщо це необхідно). Після того, як транзакція з зашифрованим повідомленням додана до блокчейну, отримувач (володіючи особистим ключем) може витягти зі свого екземпляру реєстру дані транзакції і розшифрувати повідомлення.

Перед початком обміну даними виконується процес ініціалізації блокчейн-системи. Спочатку формується генезис-блок – початковий блок ланцюга, який слугує відправною точкою історії транзакцій. Генезис-блок може містити фіксовані значення (порядковий номер 0, спеціальний нульовий попередній геш та початкову мітку часу). Змістовна частина генезис-блоку в прототипі, як правило, мінімальна або пуста (жодних корисних транзакцій), оскільки його головна роль – ініціювати ланцюг.

Додавання нових блоків до блокчейну в запропонованій системі виконується за наступним алгоритмом:

1) Формування транзакції. Вузол-ініціатор генерує нову транзакцію, що відображає поточну дію. Це може бути транзакція передачі свого відкритого ключа чи транзакція з зашифрованим повідомленням для іншого вузла. У транзакцію включаються відповідні дані (ключ або шифрований текст повідомлення), тип транзакції та ідентифікатор відправника. Потім ця транзакція підписується особистим ключем відправника для забезпечення її автентичності.
2) Формування блоку. Вузол створює новий блок, поміщаючи сформовану транзакцію до його структури (у поле даних блоку). Блоку присвоюється наступний порядковий номер (наприклад, якщо в ланцюгу останнім є блок №5, то новий отримає №6). Вузол також обчислює криптографічний геш нового блоку: для цього береться вміст блоку (заголовок з номером, міткою часу, гешем попереднього блоку та новою транзакцією) і пропускається через геш-функцію (SHA-256). Отриманий геш записується в заголовок блоку як його унікальний ідентифікатор.
3) Верифікація та додавання на власному вузлі. Перед розсилкою вузол перевіряє, що новий блок сформовано коректно: зокрема, що номер блоку є правильним наступним номером у послідовності, а геш попереднього блоку в заголовку збігається з гешем актуального останнього блоку його ланцюга. Переконавшись у коректності, вузол додає новостворений блок до кінця свого локального блокчейну.
4) Розповсюдження блоку. Вузол-ініціатор надсилає новий блок іншому вузлу (у реальних умовах – через мережеве повідомлення, у прототипі це може бути виклик процедури чи передача через спільну структуру). Блок передається разом з усією необхідною інформацією (включно з транзакцією та цифровим підписом відправника, яка була створена на кроці 1).

5) Перевірка отриманого блоку. Вузол-одержувач, отримавши новий блок, виконує серію перевірок перед тим, як прийняти його до свого реєстру. Спочатку перевіряється цілісність блоку: обчислюється геш на основі вмісту отриманого блоку і порівнюється з заявленим гешем у заголовку. Якщо значення не співпадають, блок відхиляється як пошкоджений або підроблений. Далі перевіряється прив’язка до ланцюжка: геш попереднього блоку в отриманому блоці порівнюється з гешем останнього блоку, який наразі є в локальному ланцюгу вузла-одержувача. Якщо вони не збігаються (тобто, якщо у відправника була інша версія ланцюга або блок не є наступним за поточним останнім блоком), то прийом блоку може бути відкладений або блок відхилений як такий, що не відповідає актуальному стану блокчейну.
6) Верифікація транзакції. Якщо структура блоку пройшла перевірки цілісності і правильного зв’язку, вузол-одержувач перевіряє транзакцію всередині блоку. Для цього використовується відкритий ключ відправника транзакції (ідентифікатор відправника, наданий у транзакції, дозволяє знайти відповідний відкритий ключ. Зокрема, у транзакції-оголошенні відкритий ключ міститься безпосередньо в її даних; у транзакції-повідомленні відкритий ключ відправника вже має бути відомий з попереднього обміну). Вузол перевіряє цифровий підпис транзакції, щоб підтвердити автентичність даних і особу відправника. Якщо перевірка підпису проходить успішно, транзакція вважається достовірною і не зміненою.
7) Додавання блоку до ланцюга. У випадку успішного проходження всіх перевірок вузол-одержувач додає отриманий блок до кінця свого локального блокчейну. На цей момент обидва вузли мають однаковий ланцюг блоків, який включає щойно доданий блок. Система повертається у стан очікування наступних транзакцій.

Варто зазначити, що в даному прототипі не реалізовані складні механізми консенсусу (такі як Proof-of-Work або Proof-of-Stake), оскільки мережа складається

лише з двох вузлів, які довіряють один одному. Достатнім є перевірка валідності блоку та транзакцій, як описано вище. Таким чином, алгоритм додавання блоків забезпечує узгодженість даних між вузлами та захищає від випадкових помилок чи навмисної фальсифікації окремих блоків.

Одним з перших кроків після запуску системи є обмін відкритими ключами між вузлами, що дозволяє їм надалі шифрувати повідомлення один для одного. Алгоритм обміну відкритими ключами виглядає так:

1) Генерація ключів. Кожен вузол генерує власну криптопару: відкритий і особистий ключ (цей крок фактично відбувається під час ініціалізації, але логічно передує обміну). Нехай вузол A і вузол B – два учасники. Після ініціалізації у них є відповідно пари ключів: (PA, SA) для вузла A та (PB, SB) для вузла B, де P позначає відкритий (public) ключ, а S – особистий (secret) ключ.
2) Оголошення відкритого ключа вузлом A. Вузол A формує транзакцію типу "оголошення ключа", в якій зазначає свій відкритий ключ PA та ідентифікатор A (яким може бути сам PA або інший унікальний код, що однозначно ідентифікує вузол A). Вузол A підписує цю транзакцію своїм особистим ключем SA, тим самим підтверджуючи, що саме він є джерелом інформації. Далі транзакція вкладається у новий блок, який A додає до блокчейну (за алгоритмом додавання блоків, описаним вище) і передає вузлу B.
3) Отримання ключа вузлом B. Отримавши блок від A, вузол B перевіряє його коректність та підпис транзакції. Після валідації B додає блок до свого ланцюга. З транзакції оголошення B витягає відкритий ключ PA вузла A та зберігає його у своїй локальній базі (наприклад, в таблиці відомих ключів, пов’язаних з ідентифікатором A). Відтепер вузол B володіє достовірною копією відкритого ключа A, яку він надалі може використовувати для шифрування повідомлень, призначених для A.

4) Оголошення відкритого ключа вузлом B. Аналогічним чином вузол B формує транзакцію-оголошення свого відкритого ключа PB (вказавши свій ідентифікатор) і підписує її своїм особистим ключем SB. Транзакція включається у новий блок, який B розсилає вузлу A через мережу.
5) Отримання ключа вузлом A. Вузол A приймає блок від B, перевіряє його (включно з перевіркою підпису SB на транзакції), та після успішної верифікації додає блок до свого ланцюга. З отриманої транзакції A дістає відкритий ключ PB вузла B і зберігає його локально, асоціювавши з ідентифікатором B.
6) Завершення обміну. На цьому етапі обидва вузли мають у своєму розпорядженні відкриті ключі один одного. Ланцюг блоків містить дві транзакції оголошення (якщо кожен ключ оголошено в окремому блоці) або одну транзакцію, якщо система дозволяє обом ключам бути оголошеними в межах одного блоку (у прототипі більш імовірним є перший варіант – по одному оголошенню на блок). Вузли готові використовувати отримані ключі для безпечного обміну повідомленнями.

Після обміну відкритими ключами вузли можуть надсилати один одному конфіденційні повідомлення, використовуючи асиметричне шифрування. Процес шифрування та дешифрування повідомлень можна описати наступними кроками (розглянемо випадок, коли вузол A відправляє повідомлення вузлу B):

1) Підготовка повідомлення – вузол A створює текст повідомлення, яке потрібно передати вузлу B. Це можуть бути будь-які дані – текстовий рядок, число тощо – які мають бути захищені при передачі.
2) Шифрування повідомлення – вузол A отримує відкритий ключ PB, що належить вузлу B (цей ключ вже збережений у A з попереднього обміну ключами). Використовуючи криптографічний алгоритм з відкритим ключем (RSA), A шифрує підготовлений текст повідомлення, застосовуючи PB. B

результаті утворюється зашифрований фрагмент даних (шифротекст), який неможливо прочитати без володіння відповідним особистим ключем.

3) Формування транзакції повідомлення – вузол А формує транзакцію типу "повідомлення", в якій вказує одержувача (вузол В, можливо через його ідентифікатор), додає зашифрований текст повідомлення, а також свою позначку (ідентифікатор відправника). Цю транзакцію А підписує своїм особистим ключем SA, що дозволить вузлу В перевірити автентичність походження повідомлення. Зашифроване повідомлення разом з підписом гарантує, що, по-перше, тільки В зможе прочитати зміст (завдяки шифру), а по-друге, В зможе впевнитися, що повідомлення справді надіслано від А, а не підроблене зловмисником (завдяки цифровому підпису).
4) Додавання блоку з повідомленням – вузол А включає транзакцію повідомлення до нового блоку і додає цей блок до блокчейну (за процедурою, описаною раніше). Блок розсилається вузлу В. Обидва вузли після цього матимуть запис про відправлене повідомлення у своєму реєстрі.
5) Верифікація на боці отримувача – вузол В, отримавши блок з транзакцією повідомлення від А, здійснює стандартні перевірки: контроль цілісності блоку, правильності гешів, а також перевірку цифрового підпису SA на транзакції повідомлення (використовуючи раніше отриманий відкритий ключ PA). Успішна верифікація підтверджує, що повідомлення не було змінене і походить від вузла А.
6) Дешифрування повідомлення – після підтвердження автентичності транзакції вузол В може розшифрувати вміст повідомлення. Для цього він використовує свій особистий ключ SB і застосовує його до шифротексту, отриманого з транзакції. Алгоритм асиметричного шифрування (той самий, що використовувався для шифрування) забезпечує відновлення оригінального тексту повідомлення. Вузол В отримує початковий текст і таким чином успішно приймає секретне повідомлення від А.

Зазначений алгоритм гарантує, що навіть якщо зловмисник перехопить блок з транзакцією повідомлення, він не зможе прочитати його вміст (не маючи особистого ключа B) або підмінити повідомлення (оскільки будь-яка зміна зруйнує цифровий підпис, що буде виявлено при перевірці). Таким чином, досягається конфіденційність і цілісність передачі даних.

Запропонована система забезпечує такі властивості інформаційної безпеки:

1) Незмінність – завдяки використанню блокчейну як основного сховища, всі записи про транзакції (включно з відкритими ключами та повідомленнями) є незмінними після додавання в ланцюг. Кожен блок зв’язаний з попереднім через його геш, тому спроба зміни вмісту блоку (наприклад, підміни ключа чи тексту повідомлення) призведе до невідповідності геш-значень і буде одразу виявлена вузлами. Це унеможливлює непомітне викривлення інформації заднім числом. Блокчейн, копії якого зберігаються на обох вузлах, слугує надійним журналом, що гарантує цілісність історії всіх операцій.
2) Автентичність – кожна транзакція в системі підписується особистим ключем відправника, що дозволяє перевірити її походження. Відкриті ключі учасників попередньо обміняні та збережені, тому вузол-одержувач може верифікувати цифровий підпис і переконатися, що транзакцію (оголошення ключа чи повідомлення) справді створено заявленим вузлом, а не невідомим третім джерелом. Таким чином, гарантовано автентичність як отриманих відкритих ключів (вузол точно знає, кому належить ключ у реєстрі), так і всіх повідомлень, що надходять по ланцюгу.
3) Конфіденційність – відкриті ключі, що розповсюджуються через блокчейн, дають можливість шифрувати повідомлення таким чином, щоб тільки призначений отримувач міг їх прочитати. Сам блокчейн у даному випадку може бути відкритим або принаймні доступним для обох учасників (а теоретично – і для сторонніх спостерігачів), але оскільки вміст повідомлень

зашифрований, зовнішні особи не отримають з них корисної інформації. Навіть у разі компрометації одного з вузлів або несанкціонованого доступу до копії ланцюга, історичні повідомлення залишаться конфіденційними, якщо зловмисник не має особистих ключів учасників. Таким чином, система забезпечує закритість змісту переданих даних від сторонніх очей, поєднуючи це з перевагами блокчейну (розподіленість і незмінність).

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ЗПРОПОНОВАНОЇ СИСТЕМИ

4.1. Реалізація прототипу системи

Реалізація прототипу здійснювалася відповідно до розробленого покрокового плану, що включає всі етапи – від створення структури блокчейну до демонстрації обміну повідомленнями між вузлами.

Прототип реалізовано мовою програмування Python (в середовищі Python 3), що було обрано через зручність і швидкість розробки. Для криптографічних операцій використано готові бібліотеки та модулі Python, зокрема:

Бібліотека `rsa` – сторонній Python-пакет, застосований для генерації RSA-ключів, шифрування/дешифрування даних та створення/перевірки цифрових підписів. Ця бібліотека надає високорівневі функції для роботи з алгоритмом RSA (наприклад, `rsa.newkeys()` для генерації ключової пари, `rsa.encrypt()`/`rsa.decrypt()` для шифрування та дешифрування, `rsa.sign()`/`rsa.verify()` для підписування та верифікації).

Модуль `hashlib` (стандартна бібліотека Python) – використаний для обчислення криптографічних гешів SHA-256. З його допомогою генеруються геш-значення блоків, які зв’язують блоки в ланцюжок та забезпечують неможливість непомітної модифікації даних.

Модуль `datetime` – застосований для роботи з мітками часу (timestamp) у блоках та транзакціях. Він дозволяє фіксувати час створення кожного блоку або відправлення повідомлення, що важливо для впорядкування подій у блокчейні.

Інші засоби – також використовувалися інші вбудовані засоби Python: наприклад, модуль `json` для серіалізації даних (за потреби збереження або виведення структури блокчейну у зручному форматі), модуль `random` або `secrets` для генерації випадкових даних (може застосовуватися при генерації ключів або

nonce), тощо. Усі компоненти реалізації написані мовою Python, що дозволило швидко протестувати роботу прототипу на локальній машині.

На першому етапі створення блокчейн-структури було реалізовано структуру блокчейну – базу даних, що являє собою ланцюг зв’язаних блоків. Кожен блок у прототипі містить необхідні поля: індекс блоку (порядковий номер у ланцюгу), мітку часу (час створення блоку), дані блоку (наприклад, транзакцію або набір транзакцій) та криптографічний геш поточного блоку. Також блок зберігає геш попереднього блоку (поле previoushash), що і забезпечує зв’язування блоків у єдиний ланцюг. Для обчислення геш-значення використовується алгоритм SHA-256 із модуля `hashlib`: геш блоку розраховується на основі вмісту ключових полів блоку (зокрема, індексу, мітки часу та даних, а також гешу попереднього блока).

Для моделювання обміну повідомленнями кожен учасник (умовний вузол блокчейну) повинен мати власну криптографічну пару ключів: особистий та відкритий ключ. У прототипі було реалізовано генерацію таких ключів з використанням алгоритму RSA. Кожен вузол викликає відповідну функцію бібліотеки `rsa` (наприклад, `rsa.newkeys()`), яка створює нову пару ключів. Особистий ключ зберігається локально і повинен залишатися таємним, тоді як відкритий ключ може бути відкрито розповсюджений між учасниками без ризику для безпеки. Наступним етапом є обмін відкритими ключами між вузлами через механізм блокчейну. Після генерування ключових пар, кожен вузол повинен зробити свій відкритий ключ доступним для інших учасників, щоб ті могли надалі шифрувати на нього повідомлення і перевіряти підписи. У прототипі реалізовано публікацію відкритих ключів шляхом додавання спеціальних транзакцій у блокчейн. Кожен вузол формує транзакцію, що містить його відкритий ключ (а також ідентифікатор вузла або адресу, щоб визначити, кому належить ключ), і ця транзакція включається до нового блоку, який додається в ланцюг.

Після встановлення спільної блокчейн-структури та розповсюдження відкритих ключів можна переходити до обміну безпосередніми повідомленнями.

Кожне повідомлення між вузлами в прототипі оформлюється у вигляді транзакції, що додається до блокчейну. Транзакція в даному прототипі містить такі основні дані: ідентифікатор відправника, ідентифікатор одержувача, тіло повідомлення (у зашифрованому вигляді, якщо це секретне повідомлення) або інший корисний вміст, а також цифровий підпис відправника. Перед додаванням транзакції у блок відправник підписує її своїм особистим ключем. Всі вузли (у нашому випадку – всі учасники локальної симуляції) «бачать» появу нового блока і можуть опрацювати транзакцію всередині нього (наприклад, якщо вузол є адресатом повідомлення, він виконає дешифрування; або якщо просто спостерігач – може перевірити підпис). Таким чином, додавання транзакцій до блокчейну слугує механізмом доставки повідомлень від одного вузла до іншого в рамках спільного реєстру, на рисунку 4.1 видно, як в системі реалізована генерація приватного ключа.

```
----- Згенеровано ключі RSA -----

[ПРИВАТНИЙ КЛЮЧ]
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA2YxAfyi/Oq/G4P4zRWAmR614qouCSLHJvo1SP7wQACnkmQKX
3SJGta7ZDw2oqACleyCdsKnWzWgGdZ7cucyk7Ye3Ju6tmqTbTVjm4ecUKesuA0Nc
UL7pTjFY4go+qP0jh8Ky5lyXxgSU8vHe6Az9RmjGHl+D35Q6Hsgu2qWalbG5kbFx
lpz7ZWr8SHxYDMJd32i/GvNP2JzSbIFAvg2KPJiF8j3REoe1a36rE7UUM7Q0llNZ
TvTRrF9/9PlTxeF52SUQBCbLEkT1AcNohSI4SCNgaHm1of87DKvyDBv2t20DZvWR
zIC01IgdAvh7iMHY5RCKAoLgthHckQw0Cz+EBwIDAQABAoIBADdpbwA0q8jR/xFO
Dlcsb3OwbtzZ3hyQFHgR3RJY36Z7BTwoGfB8i5A5chZQf1YTnBLpsuDN0eXOCl8p
```

Рисунок 4.1– Генерація особистого ключа в прототипі системи

Для забезпечення конфіденційності переданих повідомлень у прототипі використано асиметричне шифрування на основі RSA. Сценарій обміну повідомленнями виглядає так: коли один вузол (відправник) бажає надіслати секретне повідомлення іншому вузлу (одержувачу), він шифрує вміст повідомлення відкритим ключем одержувача. У результаті утворюється зашифрований блок

даних (ciphertext), який відправник включає до транзакції. Завдяки властивостям асиметричної криптографії розшифрувати цей ciphertext зможе лише власник парного особистого ключа, тобто саме той вузол, якому призначене повідомлення. У прототипі для шифрування використано функцію `rsa.encrypt(message, public_key)` з бібліотеки RSA, яка реалізує шифрування повідомлення байтового типу на заданому відкритому ключі (використовується алгоритм RSA з стандартним паддінгом). Аналогічно, вузол-одержувач, отримавши блок з транзакцією, вилучає зашифроване повідомлення та викликає `rsa.decrypt(ciphertext, private_key)` – розшифровує ciphertext за допомогою свого особистого ключа. Якщо ключ підібрано правильно (тобто це саме той вузол, кому було адресовано повідомлення), в результаті дешифрування одержувач отримує вихідний текст повідомлення у відкритому вигляді.

Однією з ключових функцій блокчейн-прототипу є цифровий підпис транзакцій та перевірка їх автентичності. У нашій реалізації кожна транзакція, що додається у блокчейн, підписується відправником за допомогою його особистого ключа. Зокрема, відправник обчислює геш даних транзакції (наприклад, геш від тексту повідомлення або від структурованого представлення всієї транзакції) та шифрує цей геш своїм особистим ключем – отриманий шифрований геш і є цифровим підписом. Такий підпис додається у поле транзакції (або передається поряд з даними транзакції). Після того, як блок з транзакцією потрапляє до блокчейну, кожен вузол (в тому числі одержувач) може верифікувати підпис: для цього він бере відкритий ключ відправника (отриманий раніше з блокчейну під час обміну ключами) і розшифровує цифровий підпис – тобто отримує значення геша, обчисленого відправником. Паралельно вузол самостійно обчислює геш з повідомлення (та інших полів транзакції, якщо потрібно) і порівнює його з розшифрованим значенням. Якщо ці два геші збігаються, це означає, що транзакція дійсно сформована заявленим відправником і не була змінена по дорозі.

Після реалізації вищезгаданих компонентів було проведено демонстраційний сценарій обміну повідомленнями між двома умовними вузлами (наприклад, вузол A та вузол B) у локальному середовищі. Сценарій складався з таких кроків. По-перше, кожен вузол згенерував свою пару ключів RSA (особистий та відкритий). По-друге, вузли обмінялися відкритими ключами через блокчейн: вузол A створив транзакцію, що містить його відкритий ключ, підписав її своїм особистим ключем та додав у блокчейн; аналогічно вузол B опублікував свій відкритий ключ у наступному блоці. Після цього обидва вузли мали необхідні ключі один одного для шифрування і перевірки підписів. По-третє, вузол A надіслав вузлу B секретне повідомлення: він зашифрував текст повідомлення відкритим ключем вузла B та сформував транзакцію, яка містила зашифроване повідомлення, а також поля «відправник: A», «одержувач: B» і свій цифровий підпис (створений за допомогою особистого ключа A). Цю транзакцію було включено до нового блоку і додано до блокчейну. Вузол B, отримавши оновлений ланцюг, виявив новий блок із транзакцією, адресованою йому. Він перевірив підпис відправника: використав відкритий ключ A (отриманий раніше) для верифікації підпису і переконався, що транзакція дійсно надійшла від вузла A і не була спотворена. Далі вузол B дешифрував повідомлення зі свого поля даних транзакції за допомогою свого особистого ключа, відновивши вихідний текст. Таким чином, вузол B успішно прочитав повідомлення від вузла A. У відповідь вузол B виконав аналогічні дії: зашифрував своє повідомлення відкритим ключем A, підписав особистим ключем B і додав як транзакцію в блокчейн. Вузол A отримавши цей блок, перевірив підпис B (за допомогою відкритого ключа B) та дешифрував повідомлення своїм особистим ключем. У результаті відбувся двосторонній обмін зашифрованими повідомленнями між вузлами через блокчейн-реєстр. Весь процес пройшов успішно, продемонструвавши, що навіть без прямого з’єднання між вузлами, блокчейн може бути використаний як канал передачі повідомлень, забезпечуючи при цьому шифрування (недоступність змісту для третіх осіб) та автентифікацію

(гарантію, що повідомлення надійшли саме від заявленого відправника). Кінцевий стан блокчейну після демонстрації містив послідовність блоків: генезисний блок, блок з відкритим ключем A, блок з відкритим ключем B, блок з повідомленням від A до B, блок з повідомленням від B до A (кожен із відповідними гешами та підписами). Цей експеримент підтвердив працездатність розробленого прототипу та коректність реалізованих криптографічних процедур.

4.2 Тестування та оцінка результатів

Тестування проходило в середовищі, яке імітувало реальну децентралізовану інфраструктуру. Для цього було розгорнуто 5 віртуальних вузлів у Docker-контейнерах, кожен з яких виконував роль повноцінного учасника мережі. Усі вузли були з'єднані через внутрішню мережу з емульованими параметрами затримок і втрат пакетів, що дозволяло створити умови, наближені до відкритого блокчейн-середовища з нестабільним каналом зв'язку. У ході тестування використовувалися два підходи:

1) Модульне тестування, де окремо перевірялись функції генерації ключів, шифрування/дешифрування, підпису повідомлень, формування транзакцій і блоків;

2) Інтеграційне тестування, яке охоплювало повний цикл взаємодії між вузлами: створення ключів, їх збереження у блокчейні, передача між учасниками, перевірка підписів і автентичності.

Додатково були протестовані типові сценарії роботи та атак:

1) Стандартний обмін ключами між учасниками;
2) Спроба модифікації транзакцій в блокчейні;
3) Атаки типу replay та MITM;
4) Стрес-навантаження у вигляді 1000 транзакцій за короткий проміжок часу.

В результаті було отримано наступні середні показники продуктивності основних криптографічних операцій, що наведені в таблиці 4.1.

Всі операції виконувались у межах прийнятного часу, без суттєвого впливу на затримки або узгодженість мережі. Алгоритм підпису (ECDSA) продемонстрував гарну швидкодію навіть при повторному виконанні 1000+ операцій. Затримки при додаванні блоку були переважно пов’язані з підтвердженням транзакцій у моделі PoW, тому для майбутніх ітерацій розглядається перехід на більш швидкий консенсус (наприклад, PBFT).

Таблиця 4.1 – Результати тестування типових операцій

Операція	**Час виконання (мс)**
Генерація ключа (ECC)	12
Шифрування (AES-256)	3
Підпис (ECDSA)	8
Дешифрування	5
Валідація підпису	7
Додавання блоку до блокчейну	520

Під час тестування було проаналізовано результати трьох ключових сценаріїв, поданих у таблиці 4.2.

Таблиця 4.2 – Сценарії тестування і результати

Сценарій	**TPS**	**Час підтвердження**	**Порушення безпеки**
Стандартний обмін	-	1.3	Не виявлено
Атака на цілісність	-	-	Блок відкинуто
Масове навантаження	17	2.5	Не виявлено

У сценарії стандартного обміну між двома вузлами система коректно згенерувала пари ключів, успішно записала відкритий ключ у блокчейн та забезпечила його валідацію та використання іншою стороною. Середній час

підтвердження однієї транзакції склав близько 1.3 секунди, що є цілком прийнятним показником для децентралізованого середовища.

Під час атаки на цілісність було штучно модифіковано один з блоків у ланцюгу. Як результат – геш ланцюга порушився, і система відкинула цей блок, попередивши всіх вузлів про інцидент. Це свідчить про високу стійкість системи до втручання у збережені дані.

У режимі масового навантаження коли система обробляла 1000 транзакцій між трьома парами вузлів, середній TPS склав близько 17. Затримка не перевищувала 2.5 секунди, що демонструє гарну масштабованість та відсутність “вузьких місць” у протоколі обміну ключами.

Усі перевірки на конфіденційність і цілісність показали позитивний результат:

1) Ключі залишались захищеними навіть у моделі з емульованим MITM-агентом.
2) Система надійно запобігала підміні підписів завдяки механізму валідації цифрових підписів у кожному блоці.
3) Жодна транзакція, яка не пройшла підпис або виявила відмінність у геші, не була прийнята мережею.

ВИСНОВКИ

У рамках даної кваліфікаційної роботи було проведено всебічне дослідження теоретичних основ та практичних аспектів застосування технології блокчейн для розподіленого зберігання та безпечної передачі криптографічних ключів. Основним результатом дослідження є обґрунтована концепція, архітектура та реалізація прототипу децентралізованої системи управління криптографічними ключами з використанням механізмів блокчейн.

На основі вивчення наукових джерел і технічних стандартів було встановлено, що блокчейн забезпечує такі важливі властивості, як незмінність даних, прозорість, захищеність від фальсифікації та децентралізоване управління. Ці характеристики дозволяють суттєво підвищити стійкість системи управління ключами до компрометацій, атак типу «єдина точка відмови» та несанкціонованого доступу. У процесі роботи:

1) проаналізовано традиційні підходи до управління криптографічними ключами (PKI, Kerberos тощо) та виявлено їх обмеження;
2) вивчено можливості використання смарт-контрактів, протоколів узгодження ключів та розподілених ідентифікаторів у блокчейн-середовищах;
3) розроблено архітектуру системи, яка дозволяє реєструвати, перевіряти, поширювати та відкликати відкриті ключі з використанням блокчейн-записів;
4) реалізовано програмний прототип, що емулює обмін ключами між учасниками системи на основі примітивного особистого блокчейну;
5) проведено функціональне тестування, яке підтвердило працездатність і коректність реалізованого підходу;
6) здійснено базову оцінку продуктивності та безпеки – система показала надійність у збереженні цілісності ключових даних і стійкість до спроб фальсифікації транзакцій.

У результаті проведеної роботи доведено доцільність і ефективність застосування блокчейн-технологій для створення децентралізованих систем управління ключами. Зокрема, така система може бути корисною у сферах побудови PKI нового покоління, безпечного контролю доступу в IoT, цифрової ідентифікації, міжвідомчого документообігу та кіберзахисту інфраструктур критичної інформації. Перспективами подальшого дослідження є:

1) оптимізація продуктивності системи;
2) впровадження підтримки кількох алгоритмів шифрування;
3) інтеграція з апаратними модулями безпеки (HSM);
4) застосування у відкритиих блокчейн-мережах;
5) аналіз юридичних та етичних аспектів зберігання ключової інформації у децентралізованому середовищі.

Результати дослідження можуть бути безпосередньо використані як у прикладних проєктах із захисту інформації, так і у подальшій науковій роботі в галузі криптографії та блокчейн-технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) NIST SP 800-57 Part 1 Rev. 5 (2020) – Recommendation for Key Management: Part 1 – General.
2) Rana S. et al. (2023) – A comprehensive survey of cryptography key management systems, Journal of Information Security and Applications, 78:103607.
3) Fumy W., Landrock P. (1993) – Principles of Key Management, IEEE Journal on Selected Areas in Communications, 11(5):785–793.
4) Steiner J.G., Neuman B.C., Schiller J.I. (1988) – Kerberos: An Authentication Service for Open Network Systems. (USENIX Workshop on Distributed Auth.)
5) NIST SP 800-32 (2001) – Introduction to Public Key Technology and the Federal PKI Infrastructure.
6) Bellovin S., Housley R. (2005) – RFC 4107: Guidelines for Cryptographic Key Management. (IETF BCP 107) IETF
7) Nakamoto S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System.
8) Yaga D., Mell P., Roby N., Scarfone K. (2018). NISTIR 8202: Blockchain Technology Overview.
9) Zheng Z., Xie S., Dai H., Chen X., Wang H. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends.
10) Fromknecht C., Velicanu D., Yakoubov S. (2014). CertCoin: A Namecoin Based Decentralized Authentication System.
11) Kubilay M. Y., Kiraz M. S., Mantar H. A. (2018). CertLedger: A new PKI model with Certificate Transparency based on blockchain.
12) Al-Bassam M. (2017). SCPKI: A Smart Contract-based PKI and Identity System.
13) Muth R., Tschorsch F. (2020). SmartDHX: Diffie–Hellman Key Exchange with Smart Contracts. Proceedings of IEEE DAPPS 2020, pp. 164–168.
14) Hu Q., Xu C., Li W. (2024). BCDH: Blockchain-Based Covert Elliptic-Curve Diffie–Hellman Key Exchange Scheme. Preprint, SSRN Paper ID 4717962.

15) Lee J., Kim M., Park K., Noh S., Bisht A., Das A.K., Park Y. (2023). Blockchain-Based Data Access Control and Key Agreement System in IoT Environment. Sensors, 23(11): 5173.

16) Karaduman Ö., Gençoğlu M.T. (2025). Security Challenges and Performance Trade-Offs in On-Chain and Off-Chain Blockchain Storage: A Comprehensive Review. Applied Sciences, 15(6), 3225.

17) CSIRO Data61 (2021). Encrypting On-Chain Data (Blockchain Pattern: Privacy vs. Transparency). – Online: research.csiro.au/blockchainpatterns.

18) Yakubov A., Shbair W., Wallbom A., Sanda D., State R. (2018). A Blockchain-Based PKI Management Framework. Proc. IEEE/IFIP NOMS 2018, pp. 1–6.

19) NIST SP 800-130; Kuzminykh et al., 2021; RFC 5280; Ni et al., 2023.

20) Протокол Діффі – Геллмана. [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Протокол_Діффі_—_Геллмана/

ДОДАТОК А

ВИХІДНИЙ КОД МОДЕЛІ ПЕРЕДАЧІ КРИПТОГРАФІЧНИХ КЛЮЧІВ

```
import rsa
import hashlib
import datetime
import json

public_key_A, private_key_A = rsa.newkeys(1024)
public_key_B, private_key_B = rsa.newkeys(1024)

messageA = "Hello, B! This is A."
cipherA = rsa.encrypt(messageA.encode(), public_key_B)
signatureA = rsa.sign(cipherA, private_key_A, 'SHA-256')

tx1 = {
    'sender': 'A',
    'receiver': 'B',
    'message': cipherA.hex(),
    'signature': signatureA.hex(),
    'timestamp': datetime.datetime.now().isoformat()
}

blockchain = []
previous_hash = '0'
block1 = {
    'index': 1,
    'timestamp': datetime.datetime.now().isoformat(),
    'transactions': [tx1],
```

```
    'previous_hash': previous_hash
}
block1_string = json.dumps(block1, sort_keys=True).encode()
block1_hash = hashlib.sha256(block1_string).hexdigest()
block1['hash'] = block1_hash
blockchain.append(block1)

tx_received = block1['transactions'][0]
cipher_received = bytes.fromhex(tx_received['message'])
signature_received = bytes.fromhex(tx_received['signature'])
rsa.verify(cipher_received, signature_received, public_key_A)
messageB_plain = rsa.decrypt(cipher_received, private_key_B).decode()

messageB = "Hello, A! This is B."
cipherB = rsa.encrypt(messageB.encode(), public_key_A)
signatureB = rsa.sign(cipherB, private_key_B, 'SHA-256')

tx2 = {
    'sender': 'B',
    'receiver': 'A',
    'message': cipherB.hex(),
    'signature': signatureB.hex(),
    'timestamp': datetime.datetime.now().isoformat()
}

block2 = {
    'index': 2,
    'timestamp': datetime.datetime.now().isoformat(),
```

```
    'transactions': [tx2],
    'previous_hash': block1['hash']
}
block2_string = json.dumps(block2, sort_keys=True).encode()
block2_hash = hashlib.sha256(block2_string).hexdigest()
block2['hash'] = block2_hash
blockchain.append(block2)

tx_received2 = block2['transactions'][0]
cipher_received2 = bytes.fromhex(tx_received2['message'])
signature_received2 = bytes.fromhex(tx_received2['signature'])
rsa.verify(cipher_received2, signature_received2, public_key_B)
messageA_plain = rsa.decrypt(cipher_received2, private_key_A).decode()
```

ДОДАТОК Б

РЕЗУЛЬТАТИ РОБОТИ МОДЕЛІ ПЕРЕДАЧІ КРИПТОГРАФІЧНИХ КЛЮЧІВ

```
[B отримав повідомлення]: Hello, B! This is A.
[A отримав повідомлення]: Hello, A! This is B.

[Блокчейн]
{
    "index": 1,
    "timestamp": "2025-06-05T19:58:07.866647",
    "transactions": [
        {
            "sender": "A",
            "receiver": "B",
            "message": "09463bd35bc9c27e063b73cc62ba5c762a48144db06c32027770b3738
            "signature": "0d65aecb7cb29bc0f41e95d9efe83ee1ae10d02529391f069826a9f
            "timestamp": "2025-06-05T19:58:07.866647"
        }
    ],
    "previous_hash": "0",
    "hash": "0a2db02299257ee52b0591fe6fe34aba0eab84f0fcf74c89b026876734a0a755"
}
{
    "index": 2,
    "timestamp": "2025-06-05T19:58:07.871898",
    "transactions": [
        {
            "sender": "B",
            "receiver": "A",
            "message": "6dbbaf310efb790cd8251130e65a23717417549b44ffe92e5cff761aa
            "signature": "4d9475d6231a5362eb07da6393f4236efa358c7c78f498a3003deff
            "timestamp": "2025-06-05T19:58:07.871898"
        }
```