Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий “Фізико-технічний інститут”

ЗВІТ
до лабораторної роботи №1

з дисципліни «Технологія блокчейн та розподілені системи »

Тема: «Розгортання систем Ethereum та криптовалют»

Тип роботи	Провести налаштування обраної системи та виконати тестові операції в системі Bitcoin
Виконав(ла):	Балацька Вікторія
Група:	ФІ-52мн
Перевірив(ла):	Селюх П.В

2026
м. Київ

Зміст

Вступ

Блокчейн-системи, зокрема криптовалюти та платформи виконання смарт-контрактів, є одним із найбільш значущих технологічних досягнень сучасного інформаційного суспільства. Вони являють собою розподілені інформаційні системи, що забезпечують збереження, обробку та передачу даних у децентралізованому середовищі без використання центрального органу управління. Основною особливістю таких систем є використання криптографічних методів захисту інформації, механізмів консенсусу та технології розподіленого реєстру (Distributed Ledger Technology), що гарантує цілісність, незмінність і прозорість даних.

У блокчейн-мережах всі учасники взаємодіють між собою за принципом однорангової (peer-to-peer) мережі, у якій кожен вузол зберігає копію реєстру транзакцій та бере участь у процесі перевірки нових операцій. Такий підхід забезпечує високу стійкість системи до відмов, підвищений рівень безпеки та відсутність єдиної точки контролю. Завдяки цьому блокчейн-технології широко застосовуються у фінансових системах, електронних платежах, цифровій ідентифікації, управлінні даними та інших сферах.

Однією з найбільш відомих криптовалютних систем є Bitcoin, який був запропонований у 2008 році як децентралізована платіжна система для здійснення електронних транзакцій без участі посередників. Bitcoin використовує технологію блокчейн для збереження історії транзакцій, алгоритм консенсусу Proof-of-Work для підтвердження операцій та криптографічні механізми для забезпечення автентичності й захисту даних. Особливістю даної системи є її орієнтованість на забезпечення безпечних фінансових операцій, високий рівень надійності та прозорість функціонування.

Водночас існують і більш функціонально розширені блокчейн-платформи, зокрема Ethereum, які дозволяють виконувати програмний код у мережі у вигляді смарт-контрактів та створювати децентралізовані додатки. Це визначає принципові відмінності між різними типами блокчейн-систем, їх архітектурою, механізмами роботи та можливостями застосування.

Метою даної лабораторної роботи є практичне ознайомлення з принципами функціонування криптовалютних систем, дослідження особливостей їх розгортання та налаштування, а також отримання навичок роботи з програмним забезпеченням вузла мережі Bitcoin. У межах роботи передбачено встановлення та налаштування клієнта Bitcoin Core, запуск вузла

мережі, створення криптографічного гаманця, виконання тестових транзакцій та аналіз результатів роботи системи у контрольованому середовищі.

Основними завданнями лабораторної роботи є:

- дослідження теоретичних основ функціонування блокчейн-технологій;
- ознайомлення з архітектурою та принципами роботи системи Bitcoin;
- вивчення процесу розгортання вузла криптовалютної мережі;
- виконання тестових операцій у системі;
- аналіз особливостей функціонування Bitcoin та порівняння його з іншими блокчейн-платформами.

Результати виконання лабораторної роботи дозволяють сформувати практичні навички роботи з криптовалютними системами, зрозуміти принципи їх функціонування, а також оцінити переваги та обмеження різних підходів до побудови децентралізованих інформаційних систем.

1. Теоретичні відомості

1.1. Принцип роботи Bitcoin

Bitcoin — це децентралізована електронна платіжна система, яка функціонує на основі технології блокчейн і підтримує однорангову (peer-to-peer, P2P) мережу вузлів без центрального органу управління. Основною метою системи є забезпечення можливості здійснення фінансових транзакцій між користувачами без участі посередників, таких як банки чи платіжні системи.

Архітектура Bitcoin базується на розподіленому реєстрі транзакцій, у якому всі операції зберігаються у вигляді послідовності блоків, об’єднаних у ланцюг (blockchain). Кожен блок містить набір підтверджених транзакцій, часову мітку, криптографічний хеш попереднього блоку та службову інформацію. Завдяки використанню криптографічних хеш-функцій забезпечується цілісність даних, оскільки зміна інформації в одному блоці призводить до зміни всіх наступних блоків у ланцюгу.

Надійність функціонування мережі Bitcoin забезпечується механізмом консенсусу Proof-of-Work (доказ виконаної роботи). У межах цього механізму вузли мережі, які називаються майнерами, виконують складні обчислювальні задачі з метою знаходження такого значення параметра nonce, при якому хеш заголовка блоку задовольняє встановлені вимоги складності. Процес пошуку

правильного значення nonce потребує значних обчислювальних ресурсів, що ускладнює можливість фальсифікації транзакцій і забезпечує захист мережі від атак.

Після успішного формування нового блоку він передається іншим вузлам мережі для перевірки. Якщо блок відповідає правилам протоколу, він додається до блокчейну, а інформація про нього поширюється по всій мережі. За виконання цієї роботи майнер отримує винагороду у вигляді новостворених монет Bitcoin та комісій за транзакції.

Важливою особливістю Bitcoin є використання моделі невитрачених виходів транзакцій (Unspent Transaction Output, UTXO). У межах цієї моделі баланс користувача не зберігається безпосередньо у вигляді окремого рахунку, а визначається як сукупність усіх доступних невитрачених виходів попередніх транзакцій. Кожен UTXO містить певну кількість криптовалюти та умову її витрачання, яка задається у вигляді скрипта.

Для здійснення нової транзакції користувач повинен посилатися на попередні невитрачені виходи та надати криптографічний цифровий підпис, що підтверджує право власності на відповідні кошти. Скрипт перевірки визначає умови, за яких ці кошти можуть бути витрачені. Після підтвердження транзакції використані виходи вважаються витраченими, а натомість створюються нові UTXO, які можуть бути використані у майбутніх операціях.

Захист транзакцій у системі Bitcoin базується на використанні асиметричної криптографії. Кожен користувач має пару криптографічних ключів — приватний і публічний. Приватний ключ використовується для підпису транзакцій, тоді як публічний ключ або похідна від нього адреса застосовується для отримання коштів. Такий підхід забезпечує автентичність операцій та унеможливлює несанкціонований доступ до активів користувача.

Крім того, Bitcoin використовує механізм підтвердження транзакцій, відповідно до якого операція вважається надійною лише після включення до блоку та отримання певної кількості наступних підтверджень у блокчейні. Це зменшує ризик подвійного витрачання коштів (double spending) та підвищує рівень довіри до системи.

Таким чином, принцип роботи Bitcoin базується на поєднанні розподіленої архітектури, криптографічного захисту, механізму консенсусу Proof-of-Work та

моделі UTXO, що забезпечує високий рівень безпеки, прозорості та надійності функціонування системи.

1.2. Архітектура та основні компоненти Bitcoin Core

Bitcoin Core є еталонною (reference) реалізацією протоколу Bitcoin, яка забезпечує повноцінне функціонування вузла мережі, перевірку транзакцій і блоків, підтримку локальної копії блокчейну та взаємодію з іншими вузлами через однорангову мережу. Дане програмне забезпечення реалізує основні правила протоколу Bitcoin і використовується як стандартна реалізація мережі.

Архітектура Bitcoin Core побудована за модульним принципом і включає декілька ключових компонентів, що забезпечують роботу мережевих, обчислювальних та криптографічних процесів.

Основні компоненти Bitcoin Core

1. bitcoind — демон повного вузла: bitcoind є фоновим процесом (daemon), який реалізує функціональність повного вузла мережі Bitcoin. Він забезпечує основні процеси функціонування системи, зокрема:

- встановлення та підтримку P2P-з’єднання з іншими вузлами мережі;
- прийом і передачу блоків та транзакцій
- перевірку коректності транзакцій відповідно до правил консенсусу;
- перевірку нових блоків перед їх додаванням до блокчейну;
- збереження локальної копії блокчейну;
- підтримку mempool (сховища непідтверджених транзакцій);
- забезпечення інтерфейсу віддаленого виклику процедур (RPC).

bitcoind працює у фоновому режимі та використовується переважно на серверах або у системах без графічного інтерфейсу. Саме цей компонент відповідає за основну логіку функціонування вузла мережі.

2. bitcoin-qt — графічний клієнт (GUI): bitcoin-qt є графічним інтерфейсом користувача для роботи з Bitcoin Core. Він забезпечує візуальну взаємодію з вузлом мережі та дозволяє виконувати базові операції без використання командного рядка.

Основні можливості bitcoin-qt:

- створення та керування криптографічними гаманцями;
- формування та підпис транзакцій;
- перегляд історії операцій;
- моніторинг стану синхронізації блокчейну;
- керування адресами та балансом;
- налаштування параметрів вузла.

Графічний клієнт фактично використовує ті самі механізми, що й bitcoind, але надає зручний інтерфейс для користувача.

3. bitcoin-cli — інструмент командного рядка: bitcoin-cli є інструментом командного рядка, який використовується для взаємодії з вузлом через інтерфейс RPC (Remote Procedure Call). Він дозволяє виконувати різноманітні операції керування вузлом і отримувати інформацію про його стан.

Основні функції bitcoin-cli:

- запуск RPC-команд для керування вузлом;
- отримання інформації про блокчейн;
- створення та підпис транзакцій;
- керування гаманцями;
- перевірка стану мережі;
- отримання статистики роботи вузла.

Використання bitcoin-cli є особливо важливим для автоматизації процесів, тестування та виконання лабораторних робіт.

Мережевий рівень (P2P Network Layer)

Bitcoin Core реалізує однорангову мережу, у якій вузли взаємодіють без центрального сервера. Мережевий рівень забезпечує:

- пошук і підключення до інших вузлів;
- обмін транзакціями та блоками;
- синхронізацію стану блокчейну;
- поширення нових блоків у мережі;
- захист від некоректних або шкідливих повідомлень.

Такий підхід забезпечує децентралізацію та стійкість мережі до відмов.

Mempool — сховище непідтверджених транзакцій

Mempool є тимчасовим сховищем транзакцій, які були отримані вузлом, але ще не включені до блоку. Транзакції з mempool можуть бути обрані майнерами для включення до нового блоку.

Основні функції mempool:

- збереження непідтверджених транзакцій;
- перевірка коректності транзакцій;
- управління пріоритетом транзакцій залежно від комісії;
- видалення неактуальних транзакцій.

Система зберігання даних блокчейну

Bitcoin Core підтримує локальну базу даних, яка містить:

- блоки блокчейну;
- стан UTXO;
- метадані транзакцій;
- індекси для швидкого доступу до інформації.

Локальне зберігання забезпечує незалежну перевірку транзакцій і підвищує безпеку системи.

Функції повного вузла Bitcoin

Повний вузол мережі Bitcoin виконує такі основні функції:

- перевірку всіх транзакцій відповідно до правил протоколу;
- перевірку блоків і механізму консенсусу;
- підтримку актуального стану блокчейну;
- забезпечення розповсюдження даних у мережі;
- захист від подвійного витрачання коштів;
- незалежну перевірку достовірності даних.

Запуск повного вузла підвищує рівень децентралізації мережі та забезпечує її стабільність.

Режими роботи Bitcoin Core

Для різних сценаріїв використання Bitcoin Core підтримує кілька режимів роботи мережі:

- **mainnet** — основна мережа з реальними транзакціями;
- **testnet** — тестова мережа для експериментів;
- **regtest** — локальна тестова мережа для розробки та досліджень.

У лабораторних умовах доцільно використовувати testnet або regtest, оскільки вони дозволяють виконувати експерименти без використання реальних коштів і зменшують вимоги до обчислювальних ресурсів та часу синхронізації.

Таким чином, архітектура Bitcoin Core забезпечує повний цикл функціонування криптовалютної мережі — від обробки транзакцій і перевірки блоків до збереження даних та взаємодії з користувачем. Модульна структура системи забезпечує її надійність, масштабованість та можливість використання у різних режимах роботи.

1.3. Порівняння Bitcoin та Ethereum (коротко)

Bitcoin та Ethereum є найбільш відомими блокчейн-системами, однак вони мають різне призначення, архітектуру та функціональні можливості. Незважаючи на використання спільної технологічної основи у вигляді розподіленого реєстру транзакцій, ці системи реалізують різні підходи до організації обчислень, управління станом мережі та виконання операцій.

Основною метою Bitcoin є забезпечення децентралізованої платіжної системи та збереження цифрової вартості. Система орієнтована на безпечне здійснення фінансових транзакцій між користувачами без посередників. Ethereum, у свою чергу, був створений як універсальна платформа для виконання смарт-контрактів і розробки децентралізованих додатків (DApps), що значно розширює сферу його застосування.

Однією з ключових відмінностей між Bitcoin і Ethereum є модель представлення стану системи. Bitcoin використовує модель невитрачених виходів транзакцій (UTXO), у якій баланс користувача визначається сукупністю доступних невитрачених транзакцій. Такий підхід забезпечує високий рівень безпеки та спрощує перевірку транзакцій. Ethereum застосовує модель рахунків (account-based model), де стан системи визначається балансами та даними облікових записів, що дозволяє ефективніше реалізовувати складні програмні операції.

Також системи відрізняються можливостями програмування. Bitcoin використовує обмежену скриптову мову Bitcoin Script, яка не є повноцінною мовою програмування та призначена переважно для перевірки умов витрачання коштів. Ethereum використовує віртуальну машину Ethereum Virtual Machine (EVM), яка дозволяє виконувати складні програми, написані мовами Solidity або Vyper, що забезпечує створення смарт-контрактів і децентралізованих застосунків.

Суттєвою відмінністю є механізм досягнення консенсусу. Bitcoin використовує алгоритм Proof-of-Work, який передбачає виконання складних обчислювальних задач для підтвердження блоків. Ethereum після оновлення Merge перейшов до механізму Proof-of-Stake, який базується на підтвердженні транзакцій валідаторами залежно від їх частки у системі, що зменшує енергоспоживання та підвищує ефективність роботи мережі.

Крім того, системи мають різні можливості щодо розгортання приватних мереж. У Bitcoin можливе створення локальної мережі у режимі regtest, однак така мережа не підтримує виконання складних програм або смарт-контрактів. У Ethereum можливе розгортання приватних блокчейн-мереж із повною підтримкою віртуальної машини та виконання програмного коду.

Таким чином, Bitcoin і Ethereum мають спільну технологічну основу, проте суттєво відрізняються за функціональним призначенням, архітектурою та можливостями використання. Bitcoin є спеціалізованою системою для здійснення безпечних фінансових операцій, тоді як Ethereum виступає універсальною платформою для створення програмованих децентралізованих сервісів.

2. Практична частина: налаштування системи Bitcoin

Практична частина лабораторної роботи спрямована на набуття практичних навичок розгортання та налаштування криптовалютної системи Bitcoin, а також дослідження особливостей її функціонування в реальному середовищі. На відміну від теоретичного аналізу принципів роботи блокчейн-технологій, практичне розгортання дозволяє безпосередньо дослідити процес запуску вузла мережі, створення криптографічних ключів, виконання транзакцій та перевірку їх достовірності.

У межах даної частини роботи здійснюється встановлення та налаштування програмного забезпечення Bitcoin Core, яке є еталонною реалізацією протоколу Bitcoin і забезпечує функціонування повного вузла мережі. Особлива увага приділяється запуску вузла у тестовому середовищі, що дозволяє виконувати експериментальні операції без використання реальних фінансових ресурсів та зменшує вимоги до обчислювальних ресурсів системи.

Під час виконання практичної частини досліджуються основні етапи функціонування системи Bitcoin, зокрема процес синхронізації вузла з мережею, створення та керування криптографічними гаманцями, генерація адрес для отримання коштів, виконання тестових транзакцій та аналіз результатів їх обробки. Також розглядається взаємодія вузла з іншими учасниками мережі та механізми перевірки коректності транзакцій відповідно до правил консенсусу.

Для проведення експериментів використовується тестовий режим роботи системи (testnet або regtest), який забезпечує контрольоване середовище для дослідження процесів створення блоків, підтвердження транзакцій і роботи вузла мережі. Такий підхід дозволяє детально проаналізувати особливості функціонування Bitcoin без ризику втрати реальних коштів.

Таким чином, практична частина лабораторної роботи спрямована на закріплення теоретичних знань про блокчейн-технології, формування практичних навичок роботи з криптовалютними системами та дослідження особливостей їх розгортання й експлуатації.

2.1. Вибір режиму тестування (testnet та regtest)

Для проведення лабораторної роботи з налаштування криптовалютної системи Bitcoin доцільно використовувати спеціальні тестові режими роботи мережі, які дозволяють виконувати експериментальні операції без використання реальних фінансових ресурсів. Bitcoin Core підтримує декілька режимів функціонування, серед яких основними є mainnet, testnet та regtest. У навчальних і дослідницьких цілях найчастіше застосовуються саме testnet і regtest.

Testnet є публічною тестовою мережею Bitcoin, яка функціонує аналогічно до основної мережі, але використовує тестові монети, що не мають реальної економічної вартості. Основною метою testnet є перевірка нових

функцій, тестування програмного забезпечення та дослідження поведінки системи в умовах, максимально наближених до реальної мережі.

У testnet використовуються ті самі механізми консенсусу, перевірки транзакцій і формування блоків, що і в основній мережі Bitcoin. Однак користувачі можуть отримувати тестові монети через спеціальні сервіси (faucet), що дозволяє виконувати експериментальні транзакції без фінансового ризику. До недоліків testnet належить необхідність синхронізації з мережею, що може займати значний час і потребує стабільного інтернет-з'єднання.

Regtest (Regression Test Mode) є локальною приватною мережею, яку користувач повністю контролює. У цьому режимі вузол працює ізольовано від інших мереж, що дозволяє самостійно створювати блоки, виконувати транзакції та змінювати параметри роботи системи.

Основними перевагами regtest є:

- повний контроль над процесом створення блоків;
- можливість миттєвого підтвердження транзакцій;
- відсутність необхідності синхронізації з глобальною мережею;
- мінімальні вимоги до обчислювальних ресурсів;
- можливість створення контрольованого тестового середовища.

У режимі regtest блоки можуть генеруватися вручну за допомогою спеціальних команд, що дозволяє швидко отримувати тестові монети та перевіряти роботу системи. Завдяки цьому regtest є найбільш зручним середовищем для навчальних досліджень і виконання лабораторних робіт.

У межах даної лабораторної роботи основні експерименти виконуються саме у режимі regtest, оскільки він забезпечує повністю контрольоване середовище для дослідження процесів створення блоків, підтвердження транзакцій та взаємодії вузлів мережі. Режим testnet розглядається лише з теоретичної точки зору для порівняння можливостей різних режимів функціонування системи Bitcoin.

Таким чином, використання тестових режимів дозволяє безпечно дослідити особливості функціонування криптовалютної мережі, зрозуміти принципи обробки транзакцій і підтвердження блоків, а також отримати практичні навички роботи з програмним забезпеченням Bitcoin Core.

2.2. Встановлення Bitcoin Core (Windows / Linux)

Для розгортання вузла мережі Bitcoin використовується програмне забезпечення Bitcoin Core, яке є офіційною реалізацією протоколу Bitcoin та забезпечує повноцінне функціонування вузла мережі, перевірку транзакцій, збереження блокчейну та керування криптографічними гаманцями. У даній роботі розглянуто процес встановлення Bitcoin Core для різних операційних систем: Windows, Linux та macOS.

Вибір операційної системи залежить від технічних можливостей користувача та середовища виконання лабораторної роботи. Незалежно від платформи, процес встановлення передбачає завантаження програмного забезпечення, налаштування параметрів зберігання даних та запуск вузла мережі.

2.2.1 Windows (GUI Bitcoin Core)

Для операційної системи Windows Bitcoin Core зазвичай використовується у вигляді графічного клієнта (bitcoin-qt), який забезпечує зручний інтерфейс для взаємодії з вузлом мережі.

Основні етапи встановлення:

1. Завантажити інсталятор Bitcoin Core з офіційного сайту: https://bitcoin.org
2. Запустити файл інсталяції та обрати каталог встановлення програмного забезпечення.
3. Під час першого запуску програми необхідно обрати каталог для зберігання даних блокчейну (chainstate, blocks). Для режимів testnet або regtest обсяг даних значно менший, однак каталог повинен мати достатній обсяг пам’яті та права доступу для запису.
4. Після завершення встановлення користувач може:
    - створити новий криптографічний гаманець (Create Wallet);
    - переглянути стан синхронізації мережі;
    - виконувати транзакції;
    - використовувати RPC або CLI для керування вузлом.

Графічний клієнт забезпечує повноцінну функціональність вузла та зручний доступ до основних операцій системи.

2.2.2 Linux (Ubuntu/Debian, CLI-режим)

У середовищі Linux Bitcoin Core зазвичай використовується через командний рядок, що забезпечує більшу гнучкість керування системою та можливість автоматизації процесів.

Встановлення може здійснюватися через репозиторій або шляхом завантаження офіційного архіву з сайту Bitcoin Core.

Приклад встановлення з офіційного архіву:

1. Завантажити архів із програмним забезпеченням з офіційного сайту Bitcoin Core.
2. Розпакувати архів у вибраний каталог.
3. Додати виконувані файли до системного шляху.

Приклад команд встановлення:

Розпакування архіву

tar -xvf bitcoin-*_x86_64-linux-gnu.tar.gz

Встановлення виконуваних файлів

sudo install -m 0755 -o root -g root -t /usr/local/bin bitcoin-*/bin/*

Після встановлення стають доступними основні інструменти:

- **bitcoind** — запуск вузла;
- **bitcoin-cli** — керування вузлом через RPC;
- **bitcoin-tx** — робота з транзакціями.

2.2.3 macOS (Mac OS)

Для операційної системи macOS встановлення Bitcoin Core може здійснюватися через графічний інтерфейс або за допомогою менеджера пакетів Homebrew.

Встановлення через графічний інсталятор:

1. Завантажити файл `.dmg` з офіційного сайту Bitcoin Core: https://bitcoincore.org
2. Відкрити файл інсталяції та перетягнути програму Bitcoin Core у папку Applications.
3. Запустити програму та обрати каталог для зберігання даних блокчейну.
4. Створити криптографічний гаманець або налаштувати параметри роботи вузла.

Встановлення через Homebrew (CLI-метод):

За наявності встановленого менеджера пакетів Homebrew Bitcoin Core можна встановити за допомогою командного рядка.

Послідовність дій:

Оновлення списку пакетів

brew update

Встановлення Bitcoin Core

brew install bitcoin

Після встановлення доступні інструменти:

- `bitcoind` — запуск вузла;
- `bitcoin-cli` — керування вузлом;
- `bitcoin-qt` — графічний інтерфейс.

Загальні особливості встановлення Bitcoin Core

Незалежно від операційної системи, процес встановлення Bitcoin Core передбачає:

- завантаження програмного забезпечення з офіційного джерела;
- налаштування каталогу зберігання блокчейну;
- запуск вузла мережі;
- створення криптографічного гаманця;
- синхронізацію з мережею або використання тестового режиму.

У навчальних цілях доцільно використовувати режими testnet або regtest, оскільки вони дозволяють виконувати експериментальні операції без використання реальних коштів та зменшують вимоги до системних ресурсів.

2.3. Створення конфігурації bitcoin.conf

Для зручності керування вузлом Bitcoin Core та фіксації параметрів запуску використовується конфігураційний файл `bitcoin.conf`. Він дозволяє задавати режим роботи мережі (mainnet/testnet/regtest), параметри RPC-доступу, шляхи до каталогів з даними, параметри журналювання, індексації та інші налаштування, необхідні для роботи вузла.

Використання `bitcoin.conf` є особливо важливим у лабораторних роботах, оскільки дозволяє:

- швидко перемикатися між режимами роботи (testnet/regtest);
- стабільно запускати вузол з однаковими параметрами;
- керувати вузлом через `bitcoin-cli` (RPC-інтерфейс);
- отримувати детальну діагностичну інформацію у логах;
- вмикати додаткові опції, потрібні для аналізу транзакцій і блоків.

2.3.1 Розташування конфігураційного файлу

Типове розташування файлу `bitcoin.conf` залежить від операційної системи:

- **Linux:** `~/.bitcoin/bitcoin.conf`
- **Windows:** `%APPDATA%\Bitcoin\bitcoin.conf`
- **macOS:** `~/Library/Application Support/Bitcoin/bitcoin.conf`

Якщо каталог `Bitcoin` або `.bitcoin` відсутній, його необхідно створити вручну.

2.3.2 Структура конфігурації та основні групи параметрів

Файл `bitcoin.conf` — це звичайний текстовий файл, де кожен параметр задається у форматі:

`ключ=значення`

Рядки, що починаються з `#`, є коментарями та не впливають на роботу програми. Коментарі доцільно використовувати для пояснення параметрів, особливо в лабораторній роботі.

Основні групи налаштувань у `bitcoin.conf`:

1. Налаштування мережі (режим роботи: regtest/testnet/mainnet)
2. RPC (керування вузлом через `bitcoin-cli`)
3. Логи та індексація (для аналізу)
4. Додаткові параметри продуктивності та безпеки (за потреби)

2.3.3 Приклад конфігурації для regtest (лабораторний варіант)

Нижче наведено типовий приклад конфігурації для роботи в режимі `regtest`, який є найбільш зручним для виконання лабораторних робіт.

```
# Увімкнути regtest-режим

regtest=1

# RPC (керування через bitcoin-cli)

server=1

rpcuser=student

rpcpassword=student_pass_123

rpcallowip=127.0.0.1

rpcbind=127.0.0.1

# Логи та індекси (корисно для аналізу)

debug=1
```

txindex=1

2.3.4 Пояснення параметрів конфігурації

1) `regtest=1` — увімкнення режиму regtest

Цей параметр переводить вузол у режим **локальної приватної мережі**. У regtest:

- блокчейн створюється локально (із “нуля”);
- блоки генеруються вручну командою `generatetoaddress`;
- транзакції можна підтверджувати миттєво;
- не потрібна синхронізація з інтернет-мережею Bitcoin.

Саме тому regtest є оптимальним для лабораторних робіт.

2) `server=1` — увімкнення RPC-сервера

Параметр активує можливість приймати RPC-команди. Без нього `bitcoin-cli` не зможе керувати вузлом.

Тобто якщо ти хочеш виконувати команди типу:

- `getblockchaininfo`
- `createwallet`
- `sendtoaddress`

то `server=1` має бути увімкнений.

3) `rpcuser` та `rpcpassword` — облікові дані для RPC

Ці параметри задають логін і пароль для доступу до RPC-інтерфейсу. Вони необхідні для:

- захисту вузла від стороннього доступу;
- розмежування прав доступу (базовий рівень).

У лабораторних умовах допускається простий пароль, але в реальних системах він має бути складним.

4) `rpcallowip=127.0.0.1` — дозволені IP-адреси

Цей параметр обмежує коло клієнтів, які можуть звертатися до RPC.

`127.0.0.1` означає **лише локальний комп'ютер**, що є найбезпечнішим варіантом.

5) `rpcbind=127.0.0.1` — адреса, на якій слухає RPC

Вказує, на якому інтерфейсі вузол відкриває RPC.

`127.0.0.1` також означає, що RPC буде доступний **лише локально**, а не з мережі.

6) `debug=1` — розширене журналювання (логи)

Параметр активує детальні журнали роботи вузла, що зберігаються у файлі `debug.log`.

У лабораторній роботі логи корисні, бо дозволяють:

- бачити запуск вузла;
- відслідковувати прийом транзакцій;
- аналізувати помилки конфігурації;
- перевіряти мережеві події.

7) `txindex=1` — індекс транзакцій

Цей параметр вмикає повну індексацію транзакцій. Це означає, що вузол зможе швидше і точніше відповідати на запити щодо транзакцій, які були в блокчейні.

Корисно для лабораторної, бо можна робити аналітичні RPC-запити до історичних транзакцій і блоків (особливо якщо ти робиш дослідження/перевірку транзакцій).

Отже, створення файлу `bitcoin.conf` дозволяє налаштувати Bitcoin Core під навчальні потреби: увімкнути regtest-середовище, забезпечити керування вузлом через RPC та отримати детальні логи й індексацію для подальшого аналізу тестових операцій. Це формує основу для наступних етапів

лабораторної роботи — запуску вузла, створення гаманців та виконання транзакцій.

2.4. Запуск вузла та перевірка стану

Після встановлення програмного забезпечення Bitcoin Core та створення конфігураційного файлу `bitcoin.conf` наступним етапом є запуск вузла мережі Bitcoin і перевірка його працездатності. Запуск вузла дозволяє ініціалізувати локальну копію блокчейну, активувати мережеві служби та підготувати систему до виконання транзакцій.

У межах лабораторної роботи запуск вузла здійснюється у режимі `regtest`, що забезпечує контрольоване середовище тестування та дозволяє виконувати операції без підключення до глобальної мережі Bitcoin.

2.4.1 Запуск вузла у режимі regtest

Запуск вузла здійснюється за допомогою програми `bitcoind`, яка є фоновим процесом (daemon) і реалізує основні функції повного вузла мережі.

У Linux та macOS запуск виконується через термінал, у Windows — через PowerShell, Command Prompt або графічний клієнт.

Запуск демона Bitcoin: bitcoind -daemon

Пояснення команди

- **bitcoind** — програма запуску повного вузла Bitcoin;
- **-daemon** — запуск процесу у фоновому режимі.

Після виконання цієї команди вузол починає:

- ініціалізацію бази даних блокчейну;
- запуск RPC-сервера;
- завантаження параметрів конфігурації;
- підготовку середовища для виконання транзакцій;
- створення локального блокчейну (у режимі regtest).

При успішному запуску система повертає повідомлення про запуск служби.

2.4.2 Перевірка роботи вузла

Після запуску необхідно перевірити, чи вузол працює коректно. Для цього використовується утиліта `bitcoin-cli`, яка дозволяє взаємодіяти з вузлом через RPC.

Перевірка стану блокчейну: bitcoin-cli -regtest getblockchaininfo

Призначення команди

Команда `get blockchaininfo` повертає інформацію про поточний стан блокчейну:

- назву мережі (`chain`);
- кількість блоків (`blocks`);
- стан синхронізації;
- складність мережі;
- параметри валідації;
- інші технічні характеристики.

У режимі regtest після першого запуску:

- висота блокчейну зазвичай дорівнює 0;
- синхронізація відсутня, оскільки блоки ще не створені.

Це підтверджує коректний запуск локальної мережі.

2.4.3 Перевірка мережевих параметрів

Для отримання інформації про мережеві підключення використовується команда: bitcoin-cli -regtest getnetworkinfo

Призначення команди

Команда `getnetworkinfo` дозволяє отримати:

- версію протоколу Bitcoin;
- інформацію про мережеві з’єднання;
- кількість активних підключень;

- параметри RPC;
- налаштування мережевих інтерфейсів;
- статистику переданих даних.

У режимі regtest кількість підключень зазвичай дорівнює нулю, оскільки вузол працює локально.

2.4.4 Процеси, що відбуваються під час запуску вузла

Після запуску вузла Bitcoin Core виконує ряд внутрішніх операцій:

1. зчитування конфігураційного файлу `bitcoin.conf`;
2. ініціалізація локальної бази даних блокчейну;
3. запуск RPC-сервера;
4. створення mempool для непідтверджених транзакцій;
5. підготовка криптографічних механізмів перевірки транзакцій;
6. ініціалізація журналів роботи системи.

Ці процеси забезпечують готовність вузла до виконання транзакцій та створення блоків.

2.4.5 Журнали роботи вузла (debug.log)

Після запуску вузла створюється файл журналу `debug.log`, який містить детальну інформацію про роботу системи.

Журнал дозволяє:

- перевірити правильність запуску вузла;
- виявити помилки конфігурації;
- відстежувати мережеві події;
- аналізувати процес обробки транзакцій.

Розташування файлу залежить від операційної системи:

- Linux — `~/.bitcoin/debug.log`
- Windows — `%APPDATA%\Bitcoin\debug.log`
- macOS — `~/Library/Application Support/Bitcoin/debug.log`

2.4.6 Завершення роботи вузла

Для коректного завершення роботи використовується команда: bitcoin-cli stop

Вона завершує процес вузла без пошкодження даних блокчейну.

Таким чином, запуск вузла Bitcoin у режимі regtest забезпечує створення контрольованого середовища для дослідження функціонування системи, перевірки транзакцій та генерації блоків. Використання RPC-команд дозволяє контролювати стан мережі, отримувати інформацію про блокчейн і підтверджувати правильність роботи програмного забезпечення.

2.5. Створення гаманців і адрес

Для демонстрації виконання транзакцій у лабораторній роботі доцільно створити два окремі гаманці:

- **WalletA** — відправник коштів;
- **WalletB** — отримувач коштів.

Використання двох гаманців дозволяє змоделювати реальний процес переказу Bitcoin між різними користувачами мережі.

2.5.1 Створення гаманців

Гаманці створюються за допомогою RPC-команди `createwallet`.

bitcoin-cli -regtest createwallet "WalletA"

bitcoin-cli -regtest createwallet "WalletB"

Пояснення команди:

- **bitcoin-cli** — утиліта взаємодії з вузлом через RPC;
- **-regtest** — вказує, що команда виконується у режимі regtest;
- **createwallet** — створення нового гаманця;
- `"WalletA"` — ім’я гаманця.

Після виконання команди система повертає JSON-відповідь із підтвердженням створення гаманця.

Створені гаманці зберігаються у каталозі даних вузла та доступні для подальших RPC-викликів.

2.5.2 Генерація нових адрес

Після створення гаманців необхідно отримати нові адреси для прийому коштів.

Для цього використовується команда `getnewaddress`.

ADDR_A=$(bitcoin-cli -regtest -rpcwallet=WalletA getnewaddress "A" "bech32")

ADDR_B=$(bitcoin-cli -regtest -rpcwallet=WalletB getnewaddress "B" "bech32")

Пояснення параметрів:

- **-rpcwallet=WalletA** — вказує, з яким гаманцем виконується операція;
- **getnewaddress** — створення нової адреси;
- `"A"` — мітка (label) адреси;
- `"bech32"` — тип адреси.

Після виконання команди система повертає згенеровану Bitcoin-адресу.

Для відображення адрес у терміналі:

echo $ADDR_A

echo $ADDR_B

2.5.3 Тип адреси bech32 (SegWit)

У даній лабораторній роботі використовується тип адреси **bech32**, який відповідає стандарту Segregated Witness (SegWit).

Переваги bech32:

- менший розмір транзакції;
- зниження комісії;
- покращена перевірка помилок;

- сучасний формат адреси (починається з `bcrt1` у regtest).

У режимі regtest адреси мають префікс:

bcrt1...

Це відрізняє їх від:

- mainnet — `bc1...`
- testnet — `tb1...`

2.5.4 Перевірка стану гаманців

Для перевірки інформації про гаманець використовується команда:

bitcoin-cli -regtest -rpcwallet=WalletA getwalletinfo

bitcoin-cli -regtest -rpcwallet=WalletB getwalletinfo

Команда повертає:

- баланс гаманця;
- кількість транзакцій;
- статус блокування;
- висоту останнього блоку;
- версію гаманця.

На початковому етапі баланс обох гаманців дорівнює 0 BTC.

2.5.5 Внутрішній механізм роботи гаманця

Під час створення гаманця Bitcoin Core:

1. Генерує майстер-ключ (HD wallet).
2. Формує ієрархічну структуру похідних ключів.
3. Створює криптографічні пари (приватний/публічний ключ).
4. Генерує адресу на основі публічного ключа.
5. Зберігає приватні ключі у зашифрованому вигляді.

Таким чином, кожна адреса відповідає унікальному приватному ключу, який дозволяє витрачати отримані кошти.

У результаті виконання даного етапу було:

- створено два незалежні гаманці;
- згенеровано по одній адресі для кожного гаманця;
- перевірено коректність їх функціонування;
- підготовлено середовище для демонстрації переказу коштів.

2.6. Генерація блоків у regtest та отримання тестових монет

У мережі Bitcoin нові монети створюються у процесі генерації блоків (майнінгу). У режимі `regtest` цей процес значно спрощений, оскільки блоки можуть створюватися вручну без виконання складних обчислень Proof-of-Work. Це дозволяє швидко отримувати тестові монети та виконувати експериментальні транзакції.

Під час створення блоку вузол формує спеціальну транзакцію — **coinbase-транзакцію**, яка містить винагороду майнера за створення блоку. Ця винагорода зараховується на вказану адресу гаманця.

2.6.1 Механізм coinbase та coinbase maturity

Важливою особливістю Bitcoin є те, що винагорода за створення блоку не може бути витрачена одразу. Вона стає доступною лише після підтвердження певної кількості наступних блоків.

Це правило називається **coinbase maturity**.

Основні характеристики:

- винагорода за блок становить 50 BTC у regtest (для лабораторних цілей);
- винагорода стає доступною лише після **100 підтверджень**;
- тому необхідно згенерувати щонайменше 101 блок для можливості використання монет.

Цей механізм запобігає зловживанням і забезпечує стабільність блокчейну.

2.6.2 Генерація блоків

Для генерації блоків використовується команда `generatetoaddress`, яка створює вказану кількість блоків і зараховує винагороду на задану адресу.

Генерація 101 блоку на адресу WalletA

bitcoin-cli -regtest -rpcwallet=WalletA generatetoaddress 101 "$ADDR_A"

Пояснення параметрів:

- **generatetoaddress** — команда генерації блоків;
- **101** — кількість блоків;
- **"$ADDR_A"** — адреса отримання винагороди;
- **-rpcwallet=WalletA** — використання конкретного гаманця.

Після виконання команди вузол:

1. створює нові блоки;
2. формує coinbase-транзакції;
3. зараховує винагороду на адресу гаманця;
4. додає блоки до локального блокчейну.

Команда повертає список хешів створених блоків.

2.6.3 Перевірка балансу гаманця

Після генерації блоків необхідно перевірити баланс гаманця WalletA.

bitcoin-cli -regtest -rpcwallet=WalletA getbalance

Призначення команди

Команда повертає:

- загальний баланс гаманця;
- доступні для витрачання кошти;
- підтверджені транзакції.

Після генерації 101 блоку баланс повинен містити винагороду за створення блоків.

2.6.4 Перевірка висоти блокчейну

Для перевірки кількості створених блоків використовується команда:

bitcoin-cli -regtest getblockcount

Призначення команди

Команда повертає:

- поточну висоту блокчейну;
- кількість створених блоків;
- стан локальної мережі.

Після генерації 101 блоку висота блокчейну повинна дорівнювати 101.

2.6.5 Внутрішні процеси генерації блоків у regtest

Під час генерації блоків Bitcoin Core виконує такі операції:

1. формує новий блок із coinbase-транзакцією;
2. створює хеш заголовка блоку;
3. перевіряє відповідність правилам консенсусу;
4. додає блок до локального ланцюга;
5. оновлює стан UTXO;
6. зараховує винагороду майнеру.

У режимі regtest складність мережі мінімальна, тому блоки створюються миттєво.

2.6.6 Значення генерації блоків для лабораторної роботи

Генерація блоків дозволяє:

- отримати тестові монети;
- створити початковий баланс гаманця;

- підготувати систему до виконання транзакцій;
- перевірити механізм роботи coinbase;
- дослідити процес підтвердження транзакцій.

Без цього етапу виконання подальших операцій переказу коштів є неможливим.

Висновок до етапу генерації блоків

У результаті виконання даного етапу було створено локальний блокчейн, отримано тестові монети шляхом генерації блоків та сформовано початковий баланс гаманця WalletA. Це дозволяє перейти до наступного етапу лабораторної роботи — виконання тестової транзакції між гаманцями.

3. Аналіз особливостей розгортання та обмежень

У процесі виконання лабораторної роботи було досліджено особливості розгортання криптовалютної системи Bitcoin, а також визначено її технічні обмеження, пов’язані з архітектурою системи, механізмом консенсусу та моделлю обробки транзакцій. Аналіз дозволяє оцінити вимоги до розгортання вузла, рівень безпеки системи, функціональні можливості та відмінності від інших блокчейн-платформ, зокрема Ethereum.

1) Обсяг даних та процес синхронізації

Однією з основних особливостей розгортання Bitcoin є необхідність зберігання повної копії блокчейну для забезпечення незалежної перевірки транзакцій. Повний вузол Bitcoin Core у основній мережі (mainnet) зберігає всі історичні блоки та транзакції, що потребує значного обсягу дискового простору та часу для первинної синхронізації.

Основні характеристики:

- значні вимоги до обсягу сховища даних;
- тривалий процес початкової синхронізації;
- постійне зростання розміру блокчейну;

- необхідність стабільного мережевого підключення.

У режимах testnet і regtest ці вимоги значно знижуються, оскільки:

- використовується окремий тестовий блокчейн;
- обсяг даних є невеликим;
- синхронізація відбувається швидко або не потрібна (у regtest);
- блоки можуть створюватися локально.

Таким чином, для навчальних і дослідницьких цілей використання тестових режимів дозволяє суттєво спростити процес розгортання системи.

2) Безпека та перевірка консенсусу

Bitcoin Core реалізує модель повної локальної перевірки правил консенсусу. Кожен вузол незалежно перевіряє:

- коректність транзакцій;
- криптографічні підписи;
- відсутність подвійного витрачання коштів;
- правильність формування блоків;
- відповідність правилам протоколу.

Такий підхід забезпечує високий рівень безпеки системи, оскільки вузол не довіряє зовнішнім джерелам даних і самостійно перевіряє всі операції. Децентралізований характер перевірки підвищує стійкість мережі до атак і зменшує залежність від центральних органів управління.

Однак повна валідація транзакцій потребує додаткових обчислювальних ресурсів і збільшує вимоги до апаратного забезпечення вузла.

3) Функціональність та обмеження програмування

Bitcoin спочатку проектувався як система для безпечного переказу цифрової вартості, тому його функціональність зосереджена переважно на фінансових операціях.

Bitcoin використовує мову сценаріїв **Bitcoin Script**, яка має такі особливості:

- не є Turing-complete;
- не підтримує складні цикли та умовні конструкції;
- має обмежені можливості виконання програмного коду;
- орієнтована на перевірку умов витрачання коштів.

Обмеженість мови скриптів підвищує безпеку системи, але водночас зменшує можливості реалізації складної бізнес-логіки.

На відміну від Bitcoin, платформа Ethereum підтримує повноцінну віртуальну машину (EVM), що дозволяє виконувати складні смарт-контракти та створювати децентралізовані застосунки. Таким чином, функціональні можливості Bitcoin у сфері програмування є значно обмеженими.

4) Комісії та механізм підтвердження транзакцій

У мережі Bitcoin транзакції вважаються остаточно підтвердженими лише після включення до блоку та отримання достатньої кількості наступних підтверджень. Кількість підтверджень визначає рівень довіри до транзакції.

Основні особливості:

- транзакції додаються до mempool перед включенням у блок;
- майнери обирають транзакції залежно від розміру комісії;
- підтвердження відбувається після створення нового блоку;
- у mainnet підтвердження може займати значний час.

У режимі regtest підтвердження транзакцій може бути отримано миттєво шляхом генерації нового блоку вручну. Це значно спрощує тестування та дослідження поведінки системи.

5) Неможливість прямої взаємозаміни модулів з Ethereum

Bitcoin і Ethereum мають принципово різні архітектурні підходи, що унеможливлює пряму взаємозаміну їх компонентів.

Основні відмінності:

- **модель стану:** UTXO (Bitcoin) проти account-based моделі (Ethereum);
- **механізм консенсусу:** Proof-of-Work у Bitcoin проти Proof-of-Stake в Ethereum;

- **виконання коду:** відсутність повноцінної віртуальної машини у Bitcoin;
- **архітектура транзакцій:** різні формати та правила валідації;
- **програмна логіка:** обмежений Bitcoin Script проти повноцінних смарт-контрактів Ethereum.

Через ці відмінності модулі виконання програмного коду, механізми управління станом системи та підходи до розгортання не є сумісними без використання додаткових шарів інтеграції або мостів між блокчейнами.

Отже, система Bitcoin характеризується високим рівнем безпеки та надійності, проте має значні вимоги до ресурсів при розгортанні повного вузла та обмежені можливості програмування. Водночас використання тестових режимів значно спрощує процес дослідження системи. Порівняння з платформою Ethereum демонструє принципові відмінності у архітектурі, функціональності та механізмах консенсусу, що визначає різні сфери застосування цих технологій.

4. Висновки

У ході виконання лабораторної роботи було здійснено детальне дослідження процесу розгортання та налаштування криптовалютної системи Bitcoin у тестовому середовищі. У процесі роботи було встановлено програмне забезпечення Bitcoin Core, створено конфігураційний файл `bitcoin.conf`, запущено повний вузол у режимі regtest та виконано комплекс тестових операцій, що дозволило дослідити практичні аспекти функціонування блокчейн-мережі.

У межах практичної частини лабораторної роботи було створено криптографічні гаманці та згенеровано адреси для отримання коштів, здійснено генерацію блоків для отримання тестових монет, перевірено механізм coinbase-винагороди та її доступності після відповідної кількості підтверджень. Також було досліджено процес виконання транзакцій, перевірку балансу гаманців, механізм підтвердження транзакцій та особливості роботи mempool. Отримані результати підтвердили коректність роботи вузла та дозволили на

практиці дослідити принципи функціонування розподіленого реєстру транзакцій.

У ході роботи було встановлено, що Bitcoin Core забезпечує повну локальну перевірку правил консенсусу, що підвищує рівень безпеки та надійності системи. Водночас розгортання повного вузла в основній мережі потребує значних обчислювальних ресурсів, дискового простору та часу для синхронізації. Використання тестових режимів testnet і regtest значно спрощує процес дослідження системи та дозволяє виконувати експериментальні операції без використання реальних фінансових ресурсів.

Крім того, було проведено аналіз архітектурних особливостей системи Bitcoin та її відмінностей від платформи Ethereum. Встановлено, що Bitcoin використовує модель UTXO для представлення стану системи, обмежену скриптову мову Bitcoin Script та механізм консенсусу Proof-of-Work, тоді як Ethereum застосовує account-based модель, підтримує повноцінні смарт-контракти та використовує інші механізми консенсусу. Ці відмінності визначають різні сфери застосування систем і унеможливлюють пряму взаємозаміну їх компонентів.

Отримані результати лабораторної роботи дозволили сформувати практичні навички роботи з криптовалютними системами, поглибити розуміння принципів функціонування блокчейн-технологій, механізмів підтвердження транзакцій та архітектури децентралізованих мереж. Проведене дослідження підтверджує, що Bitcoin є надійною децентралізованою платіжною системою, орієнтованою на безпечний переказ цифрової вартості, тоді як інші блокчейн-платформи можуть розширювати функціональність за рахунок підтримки програмованих смарт-контрактів.

Таким чином, поставлена мета лабораторної роботи була досягнута, а отримані результати підтверджують ефективність використання тестового середовища для дослідження особливостей розгортання та функціонування криптовалютних систем.

Список використаних джерел

1. Bitcoin Project. Bitcoin Core. Офіційний сайт: https://bitcoincore.org/
2. Bitcoin. Офіційний сайт: https://bitcoin.org/
3. Документація Bitcoin Core (RPC): https://developer.bitcoin.org/reference/rpc/
4. Medium (SWLH). How to set up a private Ethereum blockchain. (для загального контексту теми лабораторної)
5. ServerAdmin. Установка і настройка ноди Bitcoin/Ethereum та ін. (довідкові матеріали)