Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Пояснювальна записка

до магістерської кваліфікаційної роботи

«Магістр»
(освітньо-кваліфікаційний рівень)

на тему: Підвищення рівня захищеності бази даних доказової інформаційно-комунікацйної системи від несанкціонованої модифікації на основі технологій блокчейн та PoW

Виконав: ст. 2-го курсу, групи УБ-20м
спеціальності 125– Кібербезпека
Освітня програма – Управління інформаційною безпекою
(шифр і назва напряму підготовки, спеціальності)
Іванюк Т.В.
(прізвище та ініціали)
Керівник: к.т.н., доц., доцент каф. МБІС
Карпінець В.В.
(прізвище та ініціали)
«____» _____ 2021 р.
Рецензент: к.т.н., доц., доцент каф. ОТ
Войцеховська О.В.
(прізвище та ініціали)
«____» _____ 2021 р.

Вінниця ВНТУ - 2021 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Освітньо-кваліфікаційний рівень магістр
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Управління інформаційною безпекою

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

_____д.т.н., проф. Яремчук Ю.Є.
"_____"_____2021 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу студенту
_____Іванюка Тараса Володимировича_____
(прізвище, ім`я, по-батькові)

1. Тема магістерської кваліфікаційної роботи____Підвищення рівня захищеності бази даних доказової інформаційно-комунікацйної системи від несанкціонованої модифікації на основі технологій блокчейн та PoW_____
_____
Керівник магістерської кваліфікаційної роботи_к.т.н., доц., доцент каф. МБІС_
_____Карпінець В.В._____ _____
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "____"_____року №_____
2. Строк подання студентом МКР_21 грудня 2021р._____
3. Вихідні дані до МКР_____
_____
_____
_____
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)_____
_____
_____
_____
_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов`язкових креслень)
_____
_____

_____
_____

6. Консультанти розділів магістерської кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Карпінець В.В., к.т.н., доц., доц. каф. МБІС		
2	Карпінець В.В., к.т.н., доц., доц. каф. МБІС		
3	Карпінець В.В., к.т.н., доц., доц. каф. МБІС		
4	Адлер О.О., к. е. н., проф. каф. ЕПВМ		

7.Дата видачі завдання 29 вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів магістерської кваліфікаційної роботи	Примітка
	Аналіз завдання. Вступ		виконано
	Розробка технічного завдання		виконано
	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи		виконано
	Розробка алгоритму, програмна реалізація та тестування розробленого модулю захисту		
	Оформлення пояснювальної записки		
	Попередній захист та доопрацювання МКР		
	Перевірка магістерської роботи на наявність плагіату		
	Захист МКР		

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи_____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

В даній магістерській роботі було виконано підвищення рівня захищеності доказової бази даних інформаційно–комунікаційної системи від несанціонованої модифікація за рахунок програмної реалізації технологій “Proof–of–Work” та блокчейн.

У першому розділі дипломної роботи було проведено аналіз предметної області дослідження, наведено характеристики технології блокчейн. Описано властивості та практичне застосування алгоритмів консенсусу для перевірки захищеності блокчейну.

У другому розділі було розроблено алгоритм роботи системи захисту доказової бази даних інформаційно–комунікаційної системи від модифікації за рахунок використання технологій “Proof–of–Work” та блокчейн.

У третьому розділі було реалізовано модуль захисту доказової бази даних інформаційно–комунікаційної системи відповідно до вимог, вказаних у другому розділі. Наведено лістинг частин програмного коду.

Четвертий розділ дипломної роботи присвячений обгрунтуванню економічній ефективності впровадження реалізованого модулю захисту доказової бази даних інформаційно–комунікаційної системи.

ABSTRACT

В даній магістерській роботі було виконано підвищення рівня захищеності доказової бази даних інформаційно-комунікаційної системи від несанціонованої модифікація за рахунок програмної реалізації технологій “Proof-of-Work” та блокчейн.

У першому розділі дипломної роботи було проведено аналіз предметної області дослідження, наведено характеристики технології блокчейн. Описано властивості та практичне застосування алгоритмів консенсусу для перевірки захищеності блокчейну.

У другому розділі було розроблено алгоритм роботи системи захисту доказової бази даних інформаційно-комунікаційної системи від модифікації за рахунок використання технологій “Proof-of-Work” та блокчейн, .

У третьому розділі було реалізовано модуль захисту доказової бази даних інформаційно-комунікаційної системи відповідно до вимог, вказаних у другому розділі. Наведено лістинг частин програмного коду.

Четвертий розділ дипломної роботи присвячений обгрунтуванню економічній ефективності впровадження реалізованого модулю захисту доказової бази даних інформаційно-комунікаційної системи.

ЗМІСТ

ВСТУП 5

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7

1.1 Аналіз існуючих систем та засобів захисту баз даних інформаційно-комунікаційних систем 7

1.2 Характеристика технології блокчейн 12

1.3 Смарт-контракти як спосіб обміну цифровими цінностями (даними) в блокчейні 17

1.4 Використання алгоритму консенсусу для перевірки блоків 22

1.4.1 Proof-of-Work (PoW) 23

1.4.2 Proof-of-Stake (PoS) 25

1.4.3 Delegated-Proof-of-Stake (DPoS) 26

1.4.4. Proof-of-Authority (PoA) 28

1.4.5. Proof-of-Importance (PoI) 29

1.5 Висновки до першого розділу 29

2 РОЗРОБКА АЛГОРИТМУ ТА МЕТОДИ ДОСЛІДЖЕННЯ 30

2.1 Вдосконалення системи захисту баз даних від несанкціонованої модифікації доказової інформаційно-комунікаційної системи 30

2.2 Розробка алгоритму роботи підсистеми методу захисту бази даних від несанкціонованих модифікацій 33

2.3 Порівняльний аналіз розробленої підсистеми захисту бази даних . 37

2.4 Обґрунтування вибору мови програмування та засобів реалізації. . 37

2.5 Висновки до другого розділу 40

3. Практична реалізація та дослідження розробленої системи захисту 41

3.1 Програмна реалізація блокчейну 41

3.2 Програмна реалізація алгоритму Proof-of-Work 43

3.3 Створеня серверного додатку модулю захисту доказової бази даних 45

3.4 Тестування роботи розробленої системи захисту 47

3.5 Висновки до третього розділу 49

4 Економічна частина 50

4.1 Оцінювання комерційного потенціалу розробки (технологічний аудит розробки) 50

4.2 Прогнозування витрат на виконання дослідної та конструкторсько-технологічної роботи 53

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки 57

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності 59

4.5 Висновки до розділу 62

ВИСНОВКИ 63

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 64

Додаток А. Технічне завдання 66

Додаток Б. Код розробленого модулю захисту 70

ВСТУП

Однією з важливих елементів кримінального судочинства є доказова база. Її неправомірне змінення може змінити хід слідства, тому виникає необхідність забезпечення цілісності даних, та гарантування захищеності від неправомірних модифікацій протягом всього слідства. Оскільки спроби неправомірної зміни даних доказової бази були зафіксовані і з сторони співробітників, які мають безпосередній доступ, виникає необхідність застосування технологій, реалізація котрих передбачає дану можливість.

Одним з можливих варіантів вирішення даної проблеми є використання блокчейн технології - розроблена в рамках криптовалюти Bitcoin, вперше випущеної ще в 2009-му році, застосування даного механізму допоможе реалізувати децентралізованої доказової бази даних, що забезпечить прозорий механізм зберігання даних, функціонування якого не буде залежати від окремих людей.

Існує доцільність розробки інформаційно-комунікаційної системи з застосуванням технологій, які передбачають та підвищують рівень захищеності від несанкціонованої модифікації файлової складової доказової бази.

Одним з можливих засобів для підвищення захищеності може бути використання блокчейн технології для внесення та захисту файлів від несанкціонованої модифікації доказової бази.

Мета роботи полягає у підвищенні захищеності бази даних інформаційно-комунікаційної системи від несанкціонованої модифікації.

Для досягнення вищевказаної мети в роботі поставлено та реалізовано наступні задачі:

- Розглянуто інсуючі аналоги засобів та рішень для підвищення захищеності баз даних від неправомірного доступу та модифікації.
- Виконано аналіз можливісті вдосконалення існуючих методів захисту баз даних від несанкціонованої модифікації

- Розглянуто основні інструменти засобів реалізації та впровадження блокчейну. Описано переваги та недоліки, обгрунтовано вибір засобів для створення модулю захисту бази даних доказової інформаційно-комунікаційної системи.
- Створено архітектуру та алгоритм розробленого модулю захисту.
- Виконано порівняльний аналіз розробленого модулю захисту.
- Проведено аналіз найкращих рішень для реалізації клієнт-серверного застосунку.
- Реалізовано та протестовано модуль захисту бази доказової інформаційно-комунікаційної системи

Об’єкт дослідження: забезпечення захищеності від неправомірної модифікації бази даних доказової інформаційно-комунікаційної системи.

Практичне значення одержаних результатів полягає у впровадженні розробленого модулю захисту у інформаційно-комунікаційну систему та підвищить захищеність останньої від несанкціонованої модифікації.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз існуючих систем та засобів захисту баз даних інформаційно-комунікаційних систем

Розглянемо уже існуючі рішення у сфері захисту баз даних інформаційно-комунікаційних систем, застосовані інструменти, їх переваги та недоліки.

Acra-database-protection-suite

Компанія Intelligent IT Disctribution надає послуги з захищеню баз даних з централізованим типом зберігання даних. Їх продукт Acra-database-protection-suite забезпечує захист життєвого циклу даних є основою безпеки в будь-якому програмному рішенні. Acra створена для зменшення ризику витоку даних і надання зручних інструментів безпеки, що використовуються на всіх етапах життєвого циклу даних в межах програмного додатка. Acra Database Protection Suite легко інтегрувати, цей продукт не потребує значних змін в існуючому коді й забезпечує надійне шифрування та запобігання витоку інформації у ваших програмах.

Основні інструменти та характеристики які реалізовані даним програмним рішенням [1]:

- Шифрування та ключова модель: Інтеграційна бібліотека Acra шифрує дані таким чином, що дешифрувати їх можуть лише серверні компоненти Acra. Всередині рішення Acra містяться всі необхідні інструменти управління ключами для підтримки процесу: розподіл ключів, ротація, компартменталізація.
- Реалістична модель безпеки: Використання Acra допускає, що компоненти сховищ даних та програмних додатків можуть бути скомпрометовані зловмисниками, але самі дані залишаться захищеними. Acra мінімізує масштаб витоку даних, виявляє несанкціоновану поведінку та запобігає втраті даних, інформуючи операторів про інцидент, що відбувається.

- Зменшення фронту атаки: Модель шифрування Acra побудована на тому, що ніяких облікових даних, які зберігаються в компонентах додатків, не має бути достатньо для дешифрування даних, що зберігаються у backend-частині. Оскільки Acra стає єдиним шлюзом для конфіденційних незашифрованих даних, система може виконувати різні перевірки для виявлення аномальної/несанкціонованої поведінки та запису подій доступу до конфіденційних даних.
- Форм-фактор та ліцензування: Acra Open-source - ліцензована версія з відкритим ядром Apache 2. Acra Pro - покращена продуктивність, легке масштабування, інструменти управління, виділена підтримка. Acra Enterprise - покращене обслуговування, висока доступність, журнал аудиту, вбудовані функції шифрування, пошук по зашифрованих даних, підтримка SIEM. Набори опцій Acra Enterprise: спеціальні розширення для особливих випадків використання: SCADA, TimeSeries/ Моніторинг, глибока мобільна інтеграція. Acra-as-a-Service: Acra як сервіс із захищеним SQL backend на ваш вибір, попередньо налаштований та інтегрований.

Imperva SecureSphere Data Security

Ідучи в ногу із загрозою кібератак на IT сектр, забезпечувати все більш жорсткі вимоги щодо захисту та конфіденційності даних є непосильною задачею для все більшої кількості компаній, чиї засоби та програмні рішення будь-яким чином пов’язані з IT сферою. Хоча керівники та ради директорів все більше усвідомлюють ризики, реальність така, що пріортети бюджету завжди вимагають від компаній робити більше за менше фінансування. IT та безпека у сьогоденні це необхідне рішення для захисту даних, яке забезпечує безпеку, гарантію захисту і стабільність роботи бізнесу. Imperva CounterBreach і брандмауер SecureSphere Database відстежує дані та інформаційні процеси, розумно визначає та розставляє пріоритети ризиків і представляє чітку картину ризиків [2].

Imperva — найкращий вибір для захисту конфіденційних бізнес-даних і додатків, які розташовані як за допомогою хмарних технологій, так і локально. SecureSphere Database Firewall задовольняє широкий спектр вимогам

відповідності баз даних, забезпечуючи надійний захист з незначним або відсутнім впливом на продуктивність або доступність баз даних. Багаторівнева архітектура рішення масштабування для підтримки найбільших архітектурних рішень ваших програмних комплексів баз даних і рішень збереження, аналізу та доступу Big Data. За рахунок автоматизації безпеки і вимогам щодо відповідності, тисячі організацій покращили свій процес аудиту та підвищили захист своїх даних.

SecureSphere використовує два канали моніторингу – один для політик безпеки та один для аудиту. Незалежність дозволяє оптимізувати ресурси та завдання, що характерно лише для двоканального моніторингу.

Брандмауер SecureSphere Database:

- реєструє лише необхідну діяльність під час моніторингу всієї активності на предмет порушень безпеки
- Відстежує та захищає бази даних з великими обсягами транзакцій
- Блокує підозрілу поведінку для її дослідження у контексті виконуваних задач
- Виконує попередження безпеки кількома діями, усуваючи вузькі місця та затримки
- Блокує захист бази даних за допомогою брандмауера веб-додатків SecureSphere, захисту від загроз CounterBreach Insider і захисту від шкідливих програм, забезпечуючи багатофакторну безпеку даних
- Привілейований моніторинг користувачів, включаючи доступ до локального сервера
- Оновлення в режимі експлуатації мінімізують перезапуски та, як наслідок, прогалини в даних аудиту
- Гнучкість щодо вирішення проблем IT-середовища, що розвивається

McAfee Vulnerability Manager for Databases

McAfee® Vulnerability Manager для баз даних автоматично виявляє бази даних у вашій мережі; визначає, чи були встановлені останні оновлення; виконує тестування на поширені проблеми, такі як слабкі паролі, облікові записи за

замовчуванням та інші, що спрощує демонстрацію відповідності нормативним вимогам і кращий захист критичних даних.

Проводячи понад 4700 перевірок на вразливість провідних систем баз даних, таких як бази даних Oracle, Microsoft SQL Server, IBM DB2 і MySQL, McAfee Vulnerability Manager for Databases оцінює ризик практично кожного вектору загроз [3]. Але, на відміну від інших продуктів, у яких результати сканування часто можуть перевантажити вас безліччю незначних загроз, що приховують критичні проблеми, які необхідно вирішити, McAfee Vulnerability Manager для баз даних виходить далеко за межі цього. На основі даних експертів із безпеки баз даних він чітко класифікує загрози за різними рівнями пріоритету та надає сценарії виправлення, а також рекомендації. Покращуючи видимість уразливостей баз даних і надаючи експертні рекомендації щодо їх усунення, McAfee Vulnerability Manager для баз даних зменшує ймовірність шкідливого порушення та заощаджує гроші завдяки кращій підготовці до аудиту відповідності нормативним вимогам.

Завдяки набору функцій, розроблених для прискорення початкового сканування та готових звітів, щоб задовольнити більшість вимог відповідності, McAfee Vulnerability Manager для баз даних забезпечує результат, готовий до аудиту з мінімальними ресурсами.

Зламані паролі відповідають за значний відсоток порушень даних, і хакери стали набагато більш автоматизованими в своїх атаках, заснованих на простому вгадуванні паролів. Основні принципи безпеки включають уникнення слабких паролів та запобігання спільному використанню паролів між користувачами та обліковими записами, але як ви знаєте, що це відбувається? McAfee Vulnerability Manager для баз даних пропонує найшвидші доступні методи виявлення слабких паролів, позначаючи облікові записи за допомогою простих паролів, паролів за замовчуванням і спільних паролів. Він може навіть сканувати хешовані паролі, збережені, наприклад, в SHA-1, MD5 або DES. Використовуючи пряме підключення до баз даних, перевірка пароля виконується без значного

навантаження на сервер бази даних і не блокує користувачів за надмірні спроби входу.

Системи керування базами даних є складними, вони створюють власний набір ризиків безпеки, деякі з яких подібні до іншого системного програмного забезпечення (наприклад, оновлення виправлення, надійність пароля тощо), а деякі з них є унікальними для баз даних (наприклад, загрози від ін'єкції SQL). або експлойти переповнення буфера).

McAfee Vulnerability Manager для баз даних був створений командою, якій приписують участь у семи з останніх 10 критичних оновлень (CPU), які випустила Oracle. Він використовує досвід провідних спеціалістів із забезпечення безпеки баз даних для:

- Виявлення схильності до ризиків, пов'язаних із базою даних, включаючи ін'єкцію SQL, переповнення буфера та шкідливий або небезпечний код PL/SQL
- Розставленню пріоритетів та виділення «справжніх» проблем, які потребують негайної уваги
- Надання практичні відомості про те, як подолати ризики, включаючи сценарії виправлень
- Довзолу адміністраторам безпеки з обмеженими знаннями бази даних швидко зрозуміти ризики для конфіденційних даних та способи їх усунення

Переглядаючи вищенаведені програмно-комплексні рішення для забезпечення безпеки баз даних інформаційно-комунікаційних систем слід зазначити, що кожен з них має свої переваги та недоліки, але жоден з них не забезпечує захищеності від несанкціонованої модифікації людьми, які мають безпосередній доступ до взаємодії з базою даних. Саме цей фактор є ключовим у захисті доказової інформаційно-комунікаційної системи, оскільки неправомірний вплив даних осіб може бути розцінений як необхідність системою захисту, і не класифікувати юридично неправомірні дії як загрозу несанкціонованої модифікації. Саме тому доцільним є створення клієнт-

серверного модулю захисту бази даних доказової інформаційно-комунікаційної системи, використовуючи технології блокчейн та PoW.

1.2 Характеристика технології блокчейн

Технологічна модель блокчейн була розроблена в рамках криптовалюти Bitcoin. Блокчейн – це відкрита, децентралізована веб-система обліку, яка містить всі коли-небудь вчинені Bitcoin-транзакції. Ідея протоколу блокчейн полягає в тому, щоб забезпечити можливість прозорого пр оведення транзакцій між двома незнайомими сторонами без залучення центральної інстанції, які підтверджуються. Незважаючи на те, що блокчейн була розроблена для підтримки Bitcoin, дана концепція може бути визначена і використана і окремо від Bitcoin.

У загальному випадку, блокчейн – це база даних, в яку записуються факти і повні копії якої містяться на всіх комп'ютерах, об'єднаних в P2Pмережу. База даних хронологічно лінійно розширюється. Факти, які заносяться в базу, можуть бути різними, наприклад, грошові транзакції або підписи вмісту. Члени мережі – анонімні суб'єкти, що називаються вузлами мережі. Новий вузол, який приєднується до мережі, зобов'язаний завантажити повну копію блокчейн. Всі комунікації всередині мережі використовують інструменти криптографії для точної і безпечної ідентифікації відправника і одержувача. У разі, якщо вузол хоче додати факт в базу, в мережі повинен бути сфор мований консенсус, який визначає, де буде розміщений факт. Даний консенсус називається блоком.

Розглядаючи це визначення більш детально можна визначити, які нововведення несе за собою технологія блокчейн [4].

Самі по собі P2P-мережі аж ніяк не новинка, вони існують вже довгий час. P2P-мережі – це розподілені системи, яким необхідно вирішити одну дуже складну проблему інформатики – вирішення конфліктів (the resolution of conflicts), або узгодження (reconciliation). На той випадок, коли два несумісні факти надходять одночасно, система повинна мати правила, що визначають,

який з цих фактів вважати дійсним. У той час як в централізованих системах (за рахунок застосування централізованого консенсусу (тобто є тільки одна база даних, яка визначає дійсність транзакції)) і реляційних базах даних (за р ахунок посилкової цілісності) ця проблема вирішена, механізми, що застосовуються в таких системах, не можуть бути застосовані в P2P-мережах. Щоб гарантувати цілісність всередині P2P-мережі, необхідно ввести систему консенсусу – спосіб, щоб всі вузли мережі погодилися з порядком фактів. Блокчейн пропонує для цього алгоритм «докази роботи» (proof-of-work consensus), що використовує блоки. Цей децентралізований консенсус руйнує стару парадигму централізованого консенсусу.

Децентралізована схема, на якій заснований протокол Bitcoin, пер еносить авторитет і довіру на децентралізовану мережу і надає її вузлам можливість безперервно і послідовно включати свої транзакції в загальний блок і створювати унікальний «ланцюжок» – блокчейн.

Блоки – це спеціальний спосіб організації фактів в мережі недовірених користувачів (network of non-trusted peers). В їх основі лежить проста ідея: згрупувати факти в блоки, скласти з блоків єдиний ланцюжок, який буде реплікований по всіх вузлах мережі (Рисунок 1.1). Цей ланцюжок блоків видно всім вузлам в мережі. Кожен блок посилається на попередній, так може бути простежено походження кожного факту. Криптографія (в особі хеш-кодів) використовується для забезпечення безпеки аутентифікації джерела транзакції і усуває необхідність центрального посередника. Комбінація криптографії та технології блокчейн гарантує, що жодна з транзакцій не буде записана двічі. Факти, ще не додані в блок, називаються очікуваними (pending).

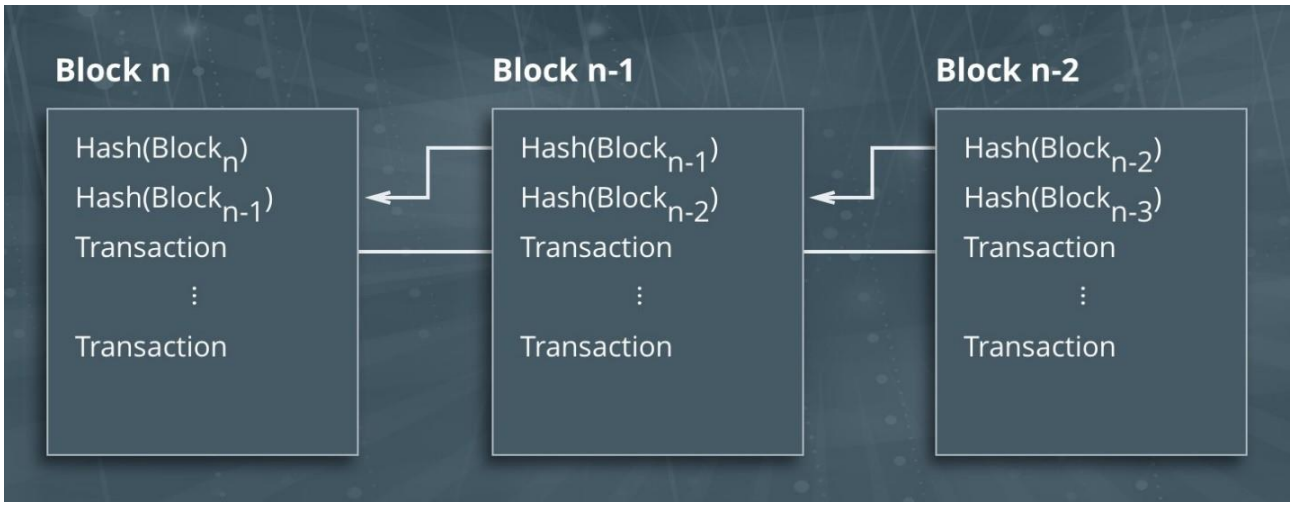

Рисунок 1.1 – Структура блокчейну

Майнінгом називається процес підтримки діяльності розподіленої платформи і «пошуку» блоків. Стандартний вузол не займається майнінгом, він лише отримує блоки від інших вузлів. Перетворення звичайного вузла у вузол - майнер – добровільне. Процес майнінгу вимагає від комп'ютер а-вузла великої обчислювальної потужності. Майнінг полягає у виборі транзакцій (фактів) з числа тих, що очікують і подальшому отриманні хеш-значення сформованого блоку транзакцій. Блок вважається успішно підтвердженим, якщо його хешзначення починається з певної кількості нулів, проте, велику частину часу отримані хеш-значення не є успішними, блок трохи змінюється і хеш-значення генерується заново. Успішно підтверджений блок включається в ланцюг і стає видимим для всіх вузлів мережі. Кількість нулів «успішного» хеш-значення вибирається таким чином, щоб кожен блок з'являвся через приблизно однакові проміжки часу. Це дуже складний процес, що важко піддається візуальному поданню. Його можна порівняти з викиданням величезної кількості гральних кісток, поки в будь-якій спробі не випадуть всі шістки. Завдяки складності даного процесу, шанс «знайти» успішне хеш-значення є вкрай низьким, а отже, запобігається можливість шахрайства (ніхто не в змозі контролювати, які саме транзакції (факти) пройдуть майнінг) і підвищується безпека мережі. Винятком є ситуація, коли шахрай володіє більш ніж половиною вузлів мережі [5].

Популярним міфом про Blockchain, який забезпечив їй безліч прихильників, так і противників, є її анонімність. З точки зору Bitcoin (і

криптовалюти в цілому), блокчейн – це ефективний інструмент для побудови розподіленої децентралізованої економіки. У той же час, існує багато можливостей для різного роду шахрайської діяльності, напр иклад, відмивання грошей.

Однак, анонімність – це єдиний спосіб захисту персональних даних користувачів. У той час, як традиційні банківські моделі надають захист даних за рахунок участі третьої особи, а в блокчейн всі транзакції (факти) видно всім вузлам. Кожен вузол може бачити відправника і одержувача, проте відсутня інформація, яка пов’язувала б транзакцію з реальними людьми [6].

Щоб дотримуватися цього принципу, наприклад, в Bitcoin рекомендується проводити кожну транзакцію з нової Bitcoin-адреси. Кількість адрес, якими може володіти користувач, не обмежена, так що дуже складно визначити, кому яка адреса належить. Однак, якщо особу відправника встановлюється, можна дуже легко простежити подальші транзакції, проведені з даної адреси.

Потенційно, блокчейн усуває будь-яких посередників при грошовому переказі, користувачеві доступний графічний інтерфейс користувача (GUI) через власний персональний комп’ютер або інший пристрій. Аналогічним функціоналом зараз оперують так звані “гаманці” для різноманітних криптовалют на ринку.

За останні кілька років було зроблено чимало спроб змінити ситуацію: прихильники підвищення рівня анонімності запропонували нові системи анонімізації (такі як TRR (Transaction Remote Release), засновану на методах шифрування, що використовуються в платформі Tor; технологія доказів з нульовим розголошенням (Zero-Knowledge Proof), коли одна зі сторін доводить наявність у неї потрібної інформації, не розголошуючи її змісту); ті, хто вважає анонімність слабким місцем даної технології, шукали методи розпізнавання відправників транзакцій.

Використання такої децентралізованої системи має свої переваги і недоліки. До переваг застосування блокчейн можна віднести [7]:

− підвищує рівень захисту від підробок за рахунок відслідковування ланцюжків походження;

− скорочує або допомагає зменшити витрати IT-інфраструктури;

− допомагає захистити величезну кількість транзакцій;

− скорочує витрати на проведення транзакцій за рахунок їх прямого проведення;

− ускладнює виникнення безконтролю за рахунок прозорості; − надає більш просту верифікацію контрольних точок даних;

− надає можливість контролю переказів усередині системи. Крім того, слід зазначити такі недоліки:

− сильна залежність від надійності криптографічних функцій;

− ускладнене регулювання через відсутність центральних адресатів;

− з ростом кількості транзакцій зростає обсяг Blockchain, і вона займає все більше місця на комп'ютерах-вузлах;

− необхідна система постійного стимулювання (вузлів-майнерів), щоб підтримувати Blockchain-інфраструктуру.

Інформація що зберігається в блокчейні існує за принципом загальнодоступної та постійно обновлюваної бази даних. Фактично, це новий принцип використання мережі, що має свої переваги та недоліки. База даних блокчейну не зберігається на єдиному носії – це означає що всі його записи є дійсно загальнодоступними та легко верифікованими. Не існує жодної централізованої та основної копії яку б могли викрасти та зламати. Інфор мація поширюється через тисячі комп'ютерів одночасно, уся інформація блокчейну доступна одразу всім учасникам інтернету. Технологія Блокчейн, так само як й інтернет, має свої вбудовані механізми захисту.

В основу технології блокчейн– роботу всіх механізмів закладено використання таких технологій та методів роботи і шифрування даних:

– асиметричні алгоритми шифрування або «асиметричні кр иптосистеми» (пари "приватних" та "публічних" ключів);

– хеш-функції або "хешування" даних (функції MD та SHA);

– хеш-таблиці для запису результатів хешування– операцій в блоках транзакцій (використання хеш-дерева типу “Дерево Меркла”);

– смарт-контракти (англ. Smart Contracts) – метод передачі даних (цифрових цінностей) від однієї особи до іншої;

– токени та реалізація механізму Proof of concept (POC) – доказ концепції, як методу верифікації події (затвердження угоди) в системі.

Шляхом зберігання однакових блоків інформації по всій мережі, блокчейн:

– не може бути контрольованим єдиною організацією;

– не має єдиного “вразливого” центру для атак хакерів;

– постійно оновлює інформацію так, що не існує застарілих або більш нових копій даних.

Незважаючи на істотні плюси використання технології Blockchain, застосування її в кожному окремому проекті повинно бути добре обдуманим і зваженим рішенням. У великій кількості випадків завдання, покладені в р амках проекту на Blockchain, можуть бути набагато більш надійно і ефективно виконані реляційними базами даних, такими як Oracle або MySQL. Якщо таке можливо в проекті, то використання реляційних баз даних є кращим, тому що вони пройшли крізь багато років розробки, мають одні з найбільш добре відтестованих і оптимізованих вихідних кодів в світі. Звичайно, це не робить блокчейн марною технологією. Однак, перш ніж починати проект, який базується на ній, необхідно мати абсолютно чітку ідею, як і з якої причини в проекті використовується дана технологія.

1.3 Смарт-контракти як спосіб обміну цифровими цінностями (даними) в блокчейні

Усі криптовалюти успішно застосовуються для проведення щоденних валютних транзакцій. Але, ті ж самі мережі, де вони реалізуються, можливо використовувати в цілях розподіленої роботи програмного забезпечення та розповсюдження даних за принципами анонімності та простих договорів. Для

даних цілей в блокчейні створюється спеціальний об'єкт – смарт-контракт. Такі програми записуються в мережі та залишаються дійсними назавжди. Крім того, у всіх учасників мережі залишається копія проведеної операції. При цьому роботу контракту можна зіставляти з управлінням грошовими операціями: створ енням аукціону, гри з грошовою винагородою, парі, тощо. Також, смарт-контракти відмінно пасують для автоматизації бухгалтерського обліку: контракт може записувати у собі, від кого і скільки прийшло грошей та планувати на цій основі фінансові прогнози на прибутки та втрати підприємства. Усі учасники мережі мають доступ до кількості операцій та їх призначення – блокчейн захищає від несанкціонованих та прихованих спекуляцій

Смарт-контракт – це програмний код, який містить інформацію про транзакцію (або, простіше, угоду) у форматі “якщо … тоді …”. Наприклад, “Якщо користувачем X буде занесено в систему 100 Вс, тоді він отримає 10 токенів N від користувача Y”. Токен – це внутрішня валюта компанії або особи, яка випускається та розповсюджується за методом залучення інвестицій. За дану валюту, в кожному конкретному випадку, інвестор може отримати різноманітні блага, що пропонує ініціатор компанії. Процедура проводиться на манер первинної публічної пропозиції акцій, за виключенням відсутності юридичних прав або державного регулювання поширення ICO (Initial Coin Offering, укр. “Первинне розміщення монет”, маючи на увазі новий підвид цифрової валюти в котру вкладено певний процент прав та бонусів інвесторами проекту). Якщо X та Y виконують свої зобов'язання, то кожен з них отримує попередньо визначений ресурс [8]. Якщо хтось з учасників даної процедури спробує уникнути виконання своїх зобов’язань, то угода буде вважатися не завершеною і сторони залишаться при своїх інтересах та майнових правах (рисунок 1.2):.

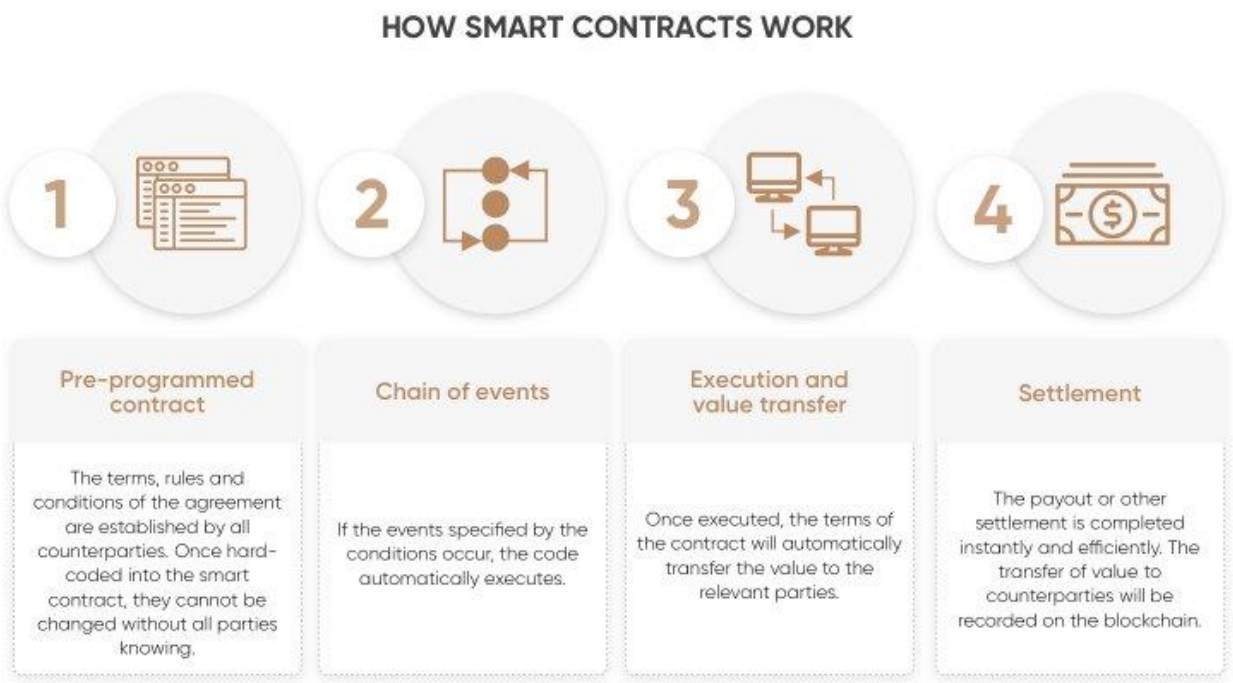

Рисунок 1.2 – Узагальнений алгоритм роботи смарт-контрактів

Принцип роботи смарт-контрактів достатньо прозорий: актив потрапляє у програму і вона сама слідкує за виконанням умов угоди. Коли вони будуть виконані, то компанія (продавець) отримає криптовалюту у встановленому еквіваленті до товару, що перейде у власність інвестора (покупця). Основними атрибутами будь-якого смарт-контракту є:

– користувачі – сторони домовленості, що приймають оговорені умови (для цього використовуються адреси облікових записів та цифрова сигнатура, або цифровий мульти-підпис, якщо підписантів контракту більше ніж два);

– об'єкт угоди– власне, ресурси для обміну (значення). При цьому, вони мають бути частиною системи, в рамках якої реалізується контракт. Якщо X та Y бажають скласти домовленість в мережі, то X повинен мати оговорену суму на рахунку, а Y – розмістити обмовлену кількість токенів в межах платформи; – умови домовленості – математично підтверджений опис умов (станів та функцій даного смарт контракту), за яких контракт буде вважатися можливим для виконання та функціональним. [9]

Залежно від блокчейн-мережі смарт-контракт може бути розроблений або на спеціалізованій мові програмування (наприклад, мова програмування Solidity, що використовується в найпоширенішій платформі для реалізації смарт-контрактів Ethereum), або на мовах загального призначення (код JavaScript, Golang, Rust, Python і тому подібних). Примітно, що термін "розумний контракт" неодноразово піддавався критиці, так як це фактично самовиконуючийся код, який алгоритмічно описує конкретні умови. У публічному блокчейні даний код видно всім учасникам мережі, які знають адресу контракту. Отже, будь-яка помилка в контракті, що веде до його уразливості, може бути використана третьою стороною. При цьому сторона, що відповідає за коректну роботу смарт-контракту часто не може оперативно виправити вразливість, так як контракт вже знаходиться в незмінному реєстрі в стані виконання. Ці умови посилюються тим, що контракти, написані Тьюринг-повних на мовах програмування типу Solidity, не можуть бути формально верифіковані (тобто правильність коду контракту щодо специфікації не може бути математично доведена і гарантована). Також, деякі блокчейн-проекти дають можливість розробляти контракти на не повних по умовах Тьюрингу мовах, але навіть при такому підході дуже складно перевірити самого блокчейн оточення, в якому виповнюється контракт. Проте, деякі блокчейни, заявляють про можливість нативної формальної верифікації.

Основні переваги смарт-контрактів мають на своїй стороні такі властивості: доказ концепції (англ. Proof of concept, PoC) є реалізацією методу або ідеєю демонстрування здійсненності або демонстрації можливості виконання домовленості в принципі з метою перевір ки того, що певна угода (інформація та дані, в широкому розумінні) мають практичний потенціал або достовірність та здійсненність. Доказ в такому випадку не обов’язково має бути повним або затвердженим в повному обсязі. Одним з найголовніших критеріїв успіху смарт-контрактів є проведення угод без залучення третьої сторони (зазвичай, в реальному світі вони виступають гарантами виконання угоди та усіх умов домовленості). Смартконтракти працюють за іншим принципом: лиш тільки реальні складові угоди потраплять в єдиний простір-платформу для обміну та

мають відповідні електронні підписи сторін угоди – домовленість вважається закритою та відбувається автоматичний обмін цінностями.

Друга перевага – безпека та конфіденційність угод. Усі контракти зберігаються в блокчейні у зашифрованому вигляді. Про умови та предмет угоди знають тільки сторони домовленості, а зробити несанкціоновані зміни у програмний код виявляється неможливим на практиці.

Третьою перевагою смарт-контрактів є зниження витрат часу та матеріальних благ на проведення угод. Якщо усі умови домовленості виконані, то користувачі здійснюють обмін активами миттєво. Але, не дивлячись на всі переваги, смарт-контракти мають і ряд правил, що мають бути виконаними для їх реалізації. Обов'язковими умовами в даному випадку виступають:

– наявність децентралізованої інформаційної середи (блокчейн), в межах якої буде знаходитись достатня кількість клієнтів для отримання та виконання запитів у всесвітньому цифровому просторі;

– наявність автоматичної бази даних (хеш-таблиці) для проведення транзакцій (укладання контрактів) в мережі;

– наявність спеціальних інструментів та алгоритмів для виконання смартконтрактів (хеш-функції), що б задовольняли вимогам безпечного розрахунку та однозначного визначення складових угоди, запису даних в БД;

– використання методів асиметричного шифрування (закритого та відкритого ключів) за допомогою яких генеруються цифрові сигнатури клієнтів мережі;

– математично доведена цілісність по Тюрінгу (можливість реалізації в системі будь-якої обчислюваної функції, що не порушує законів логіки даної системи).

Смарт-контракт – це, перш за все, програма. І, як і будь-яка програма, в ньому присутні певні недоліки:

– висока залежність від людського фактору (робота смарт-контракту та його безпечність ціликом залежать від правильності написаного програмного коду);

– недостатня адаптивність (дані, що вже занесено в блокчейн, неможливо змінити);

– погане масштабування. При одночасному запуску декількох контр актів пропускна здатність системи знижується пропорційно. Також, знову слід зазначити, що використання смарт-контракту, як і здійснення будь-якої операції у блокчейні мають бути перевірені учасниками мережі. Користувач завжди сплачує за розрахунки (перевірку хешу) незалежно від того, чи була успішною ваша операція, чи ні. Навіть якщо операція нездійсненна, майнери мають перевірити та виконати таку транзакцію (розрахувати) і саме за це має сплачувати ініціатор. Механізм дуже схожий до того, що відбувається з будь-якими банківськими операціями. Користувач може переглянути прикріплені ‘чайові’ (gas limit × gas price) до утвореної транзакції з конвертацією по курсу. Саме ця сума буде винагородженням для майнерів, що роблять роботу з розрахунків та розміщують такі операції в блоках та захищають постійність блокчейну. [10]

1.4 Використання алгоритму консенсусу для перевірки блоків

Дані у блокчейні повинні бути цілісні та добре захищені від зловмисників. Алгоритми консенсусу якраз виконують такі функції, тому вони є чи не найважливішим елементом технології блокчейн. Оскільки дані у блокчейні розподілені і немає якогось одного серверу, розподілені учасники системи повинні якось узгоджувати валідацію транзакцій, що надходять до мережі. Важливо відрізняти алгоритм консенсусу від поняття протоколу . Протокол описує правила, за якими працює система – як повинні взаємодіяти учасники мережі, які дані вони можуть передавати, які вимоги до успішної валідації блоків. У той же час, алгоритм виконує роль механізму, який перевіряє, що правила встановлені протоколом, виконуються – він валідує баланси та підписи, що підтверджують транзакції, а також фактично виконує перевірку блоків. Тобто, Bitcoin та Ethereum – це протоколи, а Proof-of-Work – це алгоритм (Рисунок 1.1) .

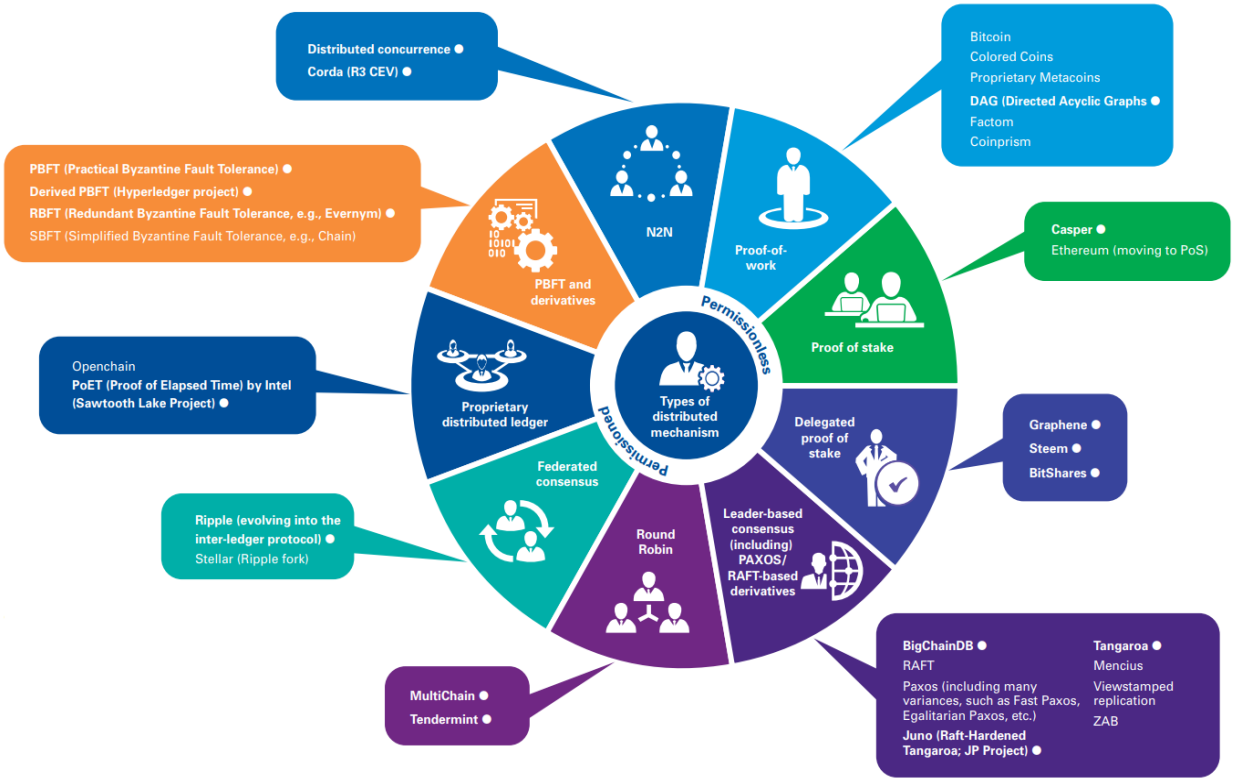

Рисунок 1.1 – Протоколи та їх застусування у сфері криптовалют

1.4.1 Proof-of-Work (PoW)

Один з найпопулярніших консенсусів, адже почав використовуватися ще у Біткоіні. Насправді, концепція Proof-of-Work була описана ще у 1993 році в роботі «Pricing via Processing, Or, Combatting Junk Mail, Advances in Cryptology» авторства Синтії Дворк та Моні Наор. Хоча термін тоді ще не був введений, автори запропонували ідею того, що для того аби отримати доступ до загального ресурсу, користувач повинен обчислити достатньою складну, але обчислювальну задачу, аби запобігти зловживанням ресурсів. Сам же термін з'явився у 1999 році в статті «Proofs of Work and Bread Pudding Protocols» авторства Маркуса Якобссона та Арі Джуелс.

Визначальними компонентами систем PoW є майнинг і електроенергія, яку вони витрачають для виконання розрахунків, що підтверджують транзакції криптовалют. Тобто суть концепції така, що усім майнерам дається задача, яку вони повинні порахувати за певний проміжок часу (у мережі Bitcoin цей час

становить 15 приблизно 10 хвилин). Задача - «Знайти таке значення x, щоб хеш SHA (x) містив N старших нульових біт». У мережі Bitcoin час вирішення задачі більш менш сталий, тому що кількість біт яку треба вирахувати динамічна і залежить від кількості учасників. Функція, що вираховується – SHA-256..

Майнер використовують комп'ютерне обладнання для запуску мережевих вузлів, які використовують обчислювальні потужності для алгоритмічного вирішення математичних головоломок, які підтримують мережу. Майнер, який вирішує головоломку першим тим самим підтверджує самий останній блок транзакції в блокчейні. Потім даний майнер передає новостворений блок всім іншим вузлам, які, в свою чергу, підтверджують його точність і додають цей блок в свою копію блокчейна. Цей процес перевірки являє собою консенсус. Тільки після підтвердження цих даних в мережу може бути доданий новий блок. Майнер отримує винагороду, як правило у вигляді криптовалюти, мережу якої він і підтримує, за те, що першим обрахував новий блок даних і додав його в блокчейн PoW.

Ланцюжки блоків, що підтверджують працездатність, націлені на створення блоків з постійними інтервалами - біткоін, наприклад, генерує один блок приблизно кожні десять хвилин. Мережі PoW обмежені з точки зору їх швидкості і масштабу, оскільки процес підтвердження роботи є енергозатратний. Більш того, Мережі PoW кодуються так, щоб бути більш-менш складними в порівнянні з обсягом обчислювальної потужності в мережі. Незважаючи на обмеження швидкості і масштабованості, блокчейни PoW історично забезпечували кращу безпеку, зберігаючи при цьому значну децентралізацію. Оскільки системи PoW розподілені, зловмиснику надзвичайно дорого захопити блокчейн, використовуючи атаку « 51% », для проведення якої необхідно мати більшу частину обчислювальної потужності в мережі.

1.4.2 Proof-of-Stake (PoS)

POS є другим за популярністю механізмом консенсусу і усуває багато недоліків, виявлені в блокчейнах PoW, такі як недостатня швидкість, погана масштабованість, неефективне споживання енергії і високий бар'єр для входу. Прикладами сучасних провідних в галузі блокчейнів POS є Polkadot, EOSIO і Cardano. Ethereum, який спочатку був розроблений як блокчейн PoW, знаходиться в процесі переходу на блокчейн POS під назвою Ethereum 2.0.

Замість майнерів, які перевіряють транзакції, в блокчейнах POS використовуються валідатори. Валідатори – це оператори мережевих вузлів, які перевіряють дані, аналогічно системам PoW, але для отримання права на перевірку не потрібно енергоємного обчислювального процесу. Замість того, щоб працювати над доказами роботи, валідатори "ставлять" деякі власні токени блокчейна, щоб отримати право на вибір в якості вузла валідатора. Потенційний валідатор, по суті, буде використовувати крипто-токени, вбудовані в блокчейн, в якості забезпечення. Коли приходить час перевірити дані, що зберігаються в блоці транзакцій в блокчейне PoS, система випадковим чином вибирає валідатор для підтвердження даних. У той час як в деякій мірі випадкові, змінні можуть підвищити ймовірність вибору валідатора, включаючи кількість токенів, які поставив валідатор. Коли блок підтверджений, цей валідатор зазвичай винагороджується комісією з транзакції, і процес починається з нового блоку.

Якщо валідатори діють зловмисно або некомпетентно, вони втрачають свою частку і доступ до мережі за допомогою процесу, званого “скорочення". Ця структура стимулів гарантує, що валідатори отримають більше вигоди від законної діяльності, ніж від порушення правил. Існує велика купа різних варіантів того, як працює цей алгоритм перевірки валідаторів.

Оскільки валідаторам на блокчейнах POS не потрібно інвестувати в дороге обладнання і високі витрати на електроенергію, бар'єр для входу в блокчейни POS для валідаторів, можливо, нижче. Однак, якщо ви хочете стати валідатором, у вас все одно має бути достатня кількість криптовалюти, щоб зробити ставку.

Тобто чим більший баланс (stake), тим у більш вигідному становищі знаходиться нода. У цьому підході майнерам теж доводиться хешувати дані, але тут складність знову ж таки залежить від балансу. Ця сума варіюється в залежності від блокчейна, але може досягати значних сумм. Блокчейни POS також піддавалися критиці, оскільки ступінь впливу валідаторів в мережі часто пропорційна розміру їх частки.

З точки зору стійкості, блокчейни POS, можливо, краще підходять для навколишнього середовища, ніж мережі PoW, тому що вони споживають значно менше електроенергії (Рисунок 1.2).

Тому прихильники стверджують, що слід зосередитися на використанні механізмів консенсусу POS в майбутніх блокчейн-проектах[11].

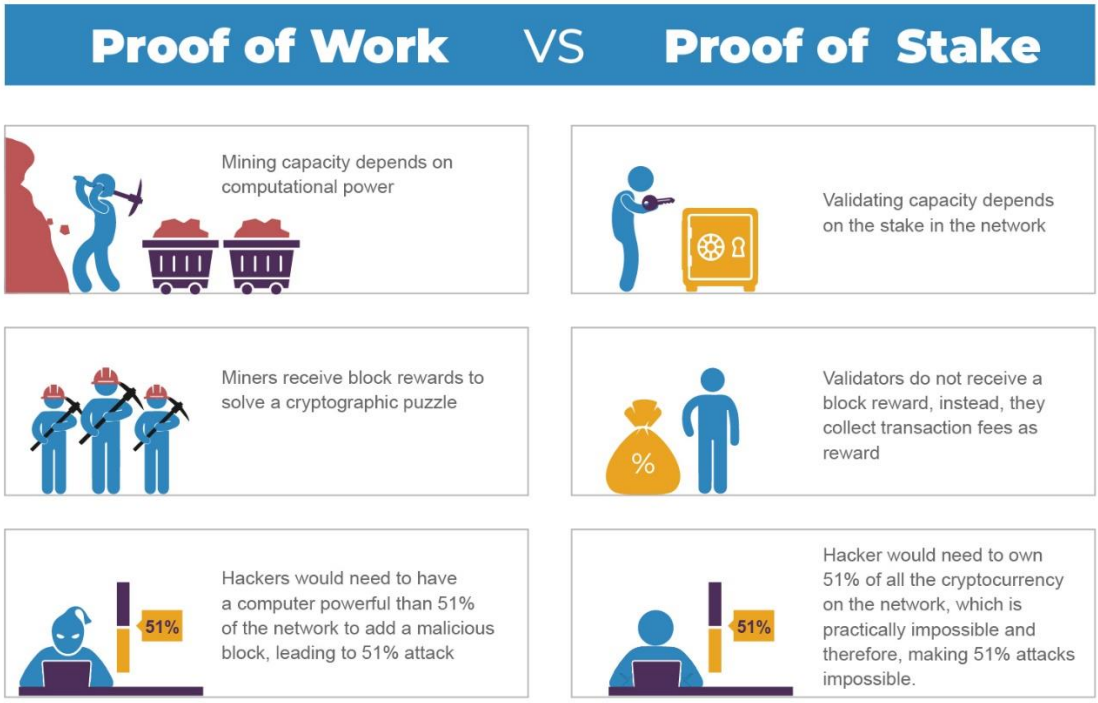

Рисунок 1.2 – Особливості алгоритмів Proof-of-Work та Proof-of-Stake

1.4.3 Delegated-Proof-of-Stake (DPoS)

Делеговане підтвердження ставки (DPoS) - популярна еволюція концепції PoS, відповідно до якої користувачі мережі голосують і вибирають делегатів для підтвердження наступного блоку. Делегатів також називають свідками або

ставорювачами блоків. Використовуючи DPO, ви можете голосувати за делегатів, об'єднавши свої токени в пул ставок і зв'язавши їх з конкретним делегатом. Ви фізично не переводите свої токени на інший гаманець, а замість цього використовуєте постачальника послуг з розміщення ставок, щоб розмістити свої токени в пулі ставок.

Для кожного нового блоку вибирається обмежене число делегатів (більшість протоколів вибирають від 20 до 100), тому делегати одного блоку можуть не бути делегатами наступного. Вибрані делегати отримують комісійні за транзакції з перевіреного блоку, і ця винагорода потім розподіляється між користувачами, які об'єднали свої токени в пул успішних делегатів. Чим більше ви ставите, тим більшу частку винагороди за блок Ви отримуєте. Нагороди розподіляються залежно від ставки кожного користувача; тому, якщо ваша ставка становить 5% від загального балансу ставок, Ви отримаєте 5% від винагороди за блок.

Перша ітерація DPoS була розроблена в 2014 році колишнім технічним директором EOS Деном Ларімером (технічний директор). Ларімер вперше впровадив алгоритм консенсусу на децентралізованій платформі криптообміну BitShares в 2015 році. Сьогодні ряд блокчейнів, включаючи Cardano, EOS і TRON, використовують DPO.

За словами його прихильників, DPoS - це більш демократичний спосіб вибору того, хто перевіряє наступний блок, дозволяючи більш різноманітній групі людей брати участь у процесі, оскільки він заснований на заслуженій репутації законного учасника, а не на загальному багатстві. Крім того, оскільки кількість валідаторів обмежена, DPoS дозволяє мережі швидше досягати консенсусу.

Триває тестування та експериментування з алгоритмами PoS, включаючи делеговані PoS. Ця концепція продемонструвала величезні перспективи для підвищення ефективності, швидкості транзакцій і пропускної здатності протоколів блокчейна, що необхідно для більш широкого використання підприємствами в міру зростання галузі і прагнення зруйнувати більш складні і великі ринки. Перехід від PoW до механізмів консенсусу на основі POS є

переломним моментом у розвитку технології блокчейна, і ітерація POS, ймовірно, стане домінуючою формою консенсусу в майбутньому.

Перевага у порівнянні з класичним Proof-of-Stake – учасники мотивовані працювати чесно, адже у будь-який момент за вас можуть перестати голосувати. До того ж, він працює швидше, ніж класичний варіант.

1.4.4. Proof-of-Authority (PoA)

Підтвердження повноважень (POA) використовує так звану модель, засновану на репутації, для перевірки транзакцій і створення нових блоків. У більшості випадків валідаторами в блокчейне консенсусу PoA є користувачі, які були обрані і схвалені іншими учасниками мережі в якості модераторів системи. В результаті валідатори, як правило, є інституційними інвесторами або іншими стратегічними партнерами в екосистемі блокчейна, які зацікавлені в довгостроковому успіху мережі і готові розкрити свою особистість заради винагороди.

Блоки записують перевірені валідатори, що завчасно обираються та по факту є модераторами системи. Тут мають цінність не кількість токенів, а репутація. Таким чином блокчейн за певним алгоритмом обирає валідатора, який запише наступний блок. Важливо зазначити, що просто так стати валідатором важко, адже треба вкласти певну кількість грошей, а також заробити довіру інших учасників мережі, аби ті голосували за нього. Але такий процес гарантує, що валідатором стане не пересічна людина.

Через процес вибору валідатора блокчейни PoA часто вважаються відносно централізованими (або" напівцентралізованими") за своєю природою. Однак той факт, що більшість блокчейнів PoA обмежують кількість валідаторів, дозволених в їх мережі, допомагає цим системам досягти високого рівня масштабованості. В результаті механізми консенсусу PoA, як правило, вважаються несумісними з повністю децентралізованими системами без дозволів, але можуть бути ефективним вибором для приватних, дозволених блокчейнів або консорціумів,

які вважають корисним публічно розкривати інформацію про своїх основних зацікавлених сторін екосистеми.

Проекти, що використовують підтвердження повноважень, включають VeChain і TomoChain.

1.4.5. Proof-of-Importance (PoI)

Proof-of-Importance наразі активно використовується у криптовалюті NEM. Цей алгоритм надає перевагу користувачам, які заслужили хорошу репутацію у мережі – «спочатку ви працюєте на репутацію, потім вона на вас». Репутація зростає з активним життям у екосистемі блокчейну та взаємодії з іншими учасниками. Чим краща репутація – тим більший шанс на створення наступного блоку . Proof-of-Importance вирішує проблему Proof-of-Stake коли один учасник або група людей мали можливість контролювати мережу, отримавши більше токенів. Тут же кількість токенів на балансі не збільшують шанси на створення блоку. До того ж, коштами треба активно користуватися, адже торгувати ними вигідніше, аніж просто тримати на балансі

1.5 Висновки до першого розділу

У першому розділі дипломної роботи було проведено аналіз предметної області дослідження, описано характеристики технології блокчейн, наведено її плюси та мінуси, розглянуто найпопулярніші алгоритми консенсусу, їх переваги і недоліки.

Розглянуто існуючі аналоги систем та засобів для підвищення захищеності баз даних інформаційно-комунікаційних систем та отримано висновок, що їх застосування не забезпечує захищеності від неправомірної модифікації людьми, які мають безпосередній доступ до бази даних.

2 РОЗРОБКА АЛГОРИТМУ ТА МЕТОДИ ДОСЛІДЖЕННЯ

Існує доволі велика вибірка реалізацій технології Blockchain, на основі яких може бути побудований клієнт-серверний додаток. У даній роботі була використана технологія блокчейн в поєднанні з технологією PoW (Proof-of-work) для підвищення захищеності бази даних доказової інформаційно-комунікаційної системи від несанкціонованої модифікації.

2.1 Вдосконалення системи захисту баз даних від несанкціонованої модифікації доказової інформаційно-комунікаційної системи

Для забезпечення системи захисту бази даних інформаційно-комунікаційної системи доцільно розробити програмний засіб у вигляді модулю, інтеграція якого матиме попередньо визначені інструменти та засоби для взаємодії з уже існуючим середовищем застосування.

Модуль захисту бази даних доказової інформаційно-комунікаційної системи повинен взаємодіяти з будь-якими змінами у доказовій базі даних та своєчасно передавати вхідні дані.

Модуль реалізований у вигляді серверної та клієнтської частин.

Серверна частина модулю захисту представляє собою програмний продукт, який складається з 6 блоків: криптографічного блоку, блоку взаємодії з базою даних доказової інформаційно-комунікаційнї системи, блоку взаємодії з вузлами мережі блокчейн, блок вирішення проблем при отримані результатів клієнтських обчислень, блок консенсусу (PoW). Архітектура розробленого модулю захисту бази даних доказової інформаційно-комунікаційної системи зображено на рисунку 2.1:

Криптографічни блок представляє собою функцію, яка спрямована на перетворення вхідних даних, а саме: хешу бази даних, часової мітки, хешу попереднього блоку у наступний блок. Результат роботи криптографічного

блоку передається на клієнтську частину для оброблення за вибраним алгоритмом консенсусу Proof-Of-Work.

Блок взаємодії з базою даних являє собою функцію, яка реагує на кожне нове підключення до бази даних доказової інформаційно-комунікаційної системи. Оскільки підключення відбувається лише тоді, коли йде допис у базу даних, тобто її передбачуваний системою запис інформації, то блок отримує нове значення хешу бази даних, і відправляє її у криптографічний блок.

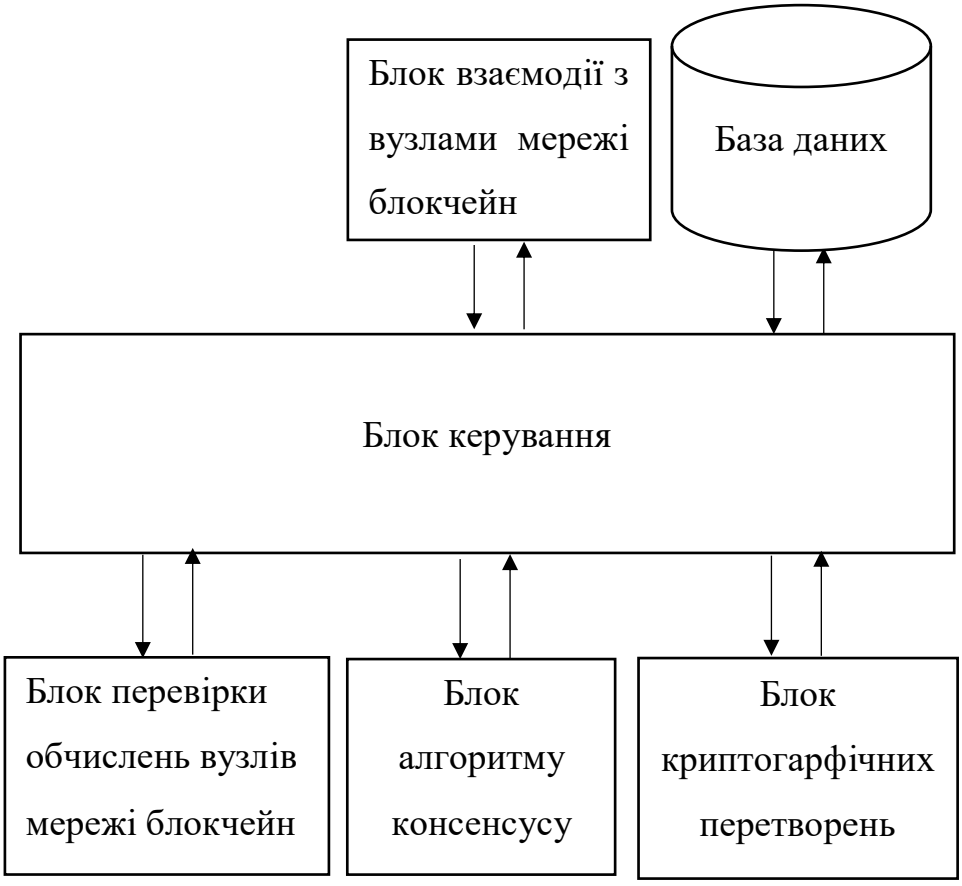

Рисунок 2.1 – Алгоритм роботи серверної частини модулю захисту

Блок взаємодії з вузлами мережі блокчейн створений для взаємодії серверної частини з децентралізованими вузлами, для обробки та створення блоків. Даний блок отримує інформацію для обробки за допомогою обраного алгоритму консенсусу Proof-Of-Work та відправляє її на сервер для додавання у блокчейн. Також вузол мережі зберігає на своїй стороні актуальну версію блокчейну.

Блок вирішення проблем при отриманні результатів децентралізованих вузлів блокчейну був створений для перевірки даних, яки надсилаються вузлами мережі блокчейну та додаються у блокчейн. Оскільки клієнтська частина може застосовуватись на засобах з різними обчислювальними потужностями, то виникає проблема у знаходженні єдиного правильного варіанту обчислення алгоритму консенсусу для блока. Для вирішення даної проблеми і застосовується даний блок.

Блок консенсусу реалізований для підтвердження достеменності результатів обробки блоків клієнтами децентралізованої мережі блокчейн.

Клієнтська частина, яка необхідна для підтримки блокчейн та використовується для створення блоків – вузли мережі складається з блоків керування, криптографічних перетворень за обраним алгоритмом консенсусу, блоку взаємодії з серверною частиною, блоку збереження кожної транзакції у вузлі блокчейну, що зображено на рисунку 2.2:

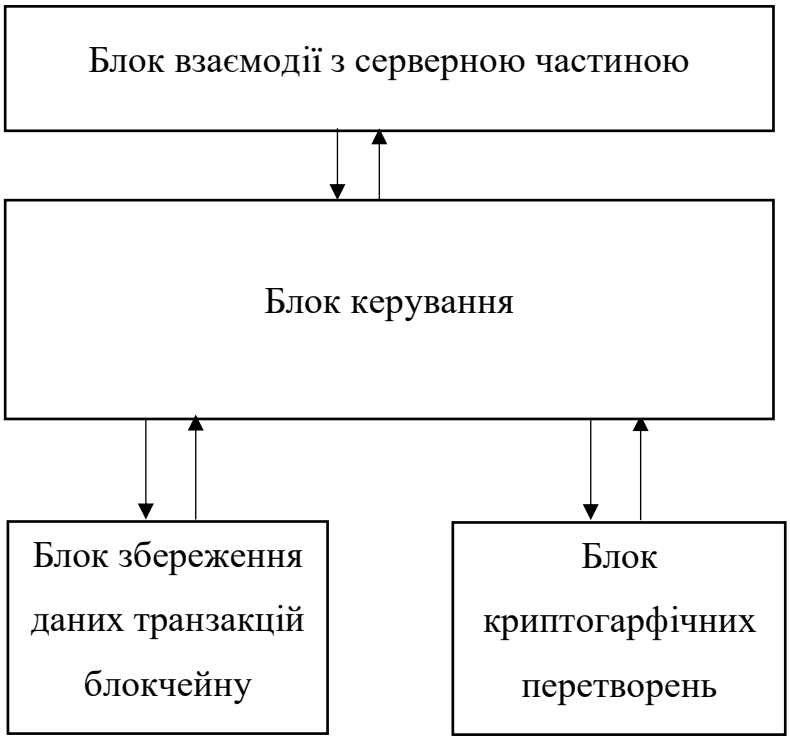

Рисунок 2.2 – алгоритм роботи клієнтського модулю захисту

Структура блоку криптографічних перетворень на клієнтській частині аналогічна відповідному блоку на серверній частині. Блок взаємодії з серверною

частиною відповідає за обмін інформацією та отриманню нових значень для обробки і відправки на сервер для утворення блокчейну. Блок збереження даних транзакцій блокчейну відповідає за збереження блоків, послідновність яких і створює блокчейн.

2.2 Розробка алгоритму роботи підсистеми методу захисту бази даних від несанкціонованих модифікацій

Узагальнений алгоритм роботи модулю захисту бази даних доказової інформаційно-комунікаційної системи наведено на рисунку 2.2.

Модуль захисту реалізований для забезпечення децентралізованого збереження даних, завдяки чому надає підвищену захищеність бази даних доказової інформацйно-комунікаційної системи, оскільки для її неправормірної модифікації, на відміну від аналогів, недостатньо мати доступ до самої бази даних, тому, що метадані та хеш суми внесених даних у інформаційно-комунікаційну систему зберігаються на вузлах мережі блокчейну.

Крок 1: Інформаційно-комунікаційна система відслідковує дію користувача і вносить дані у базу даних

Крок 2: Модуль захисту реагує на зверненя інформаційно-комунікаційної системи на створення запису у базі даних

Крок 3: Модуль захисту ініціалізує створення нового блоку, який має наступні значення:

Отримання індексу новоствореного блоку *index* шляхом збільшення значення довжини блокчейну на одиницю. Використання даного параметру вирішує проблему сумісності результатів обробки блоків на вузлах з різною швидкодією обчислювальних засобів.

Отримання значення часової мітки *timestamp*. Її застосування визначає час, коли модуль захисту ініціалізував створення блоку з точністю до хвилини.

Отримання хеш-суми введених даних *msghash*. Даний параметр забезпечує швидкодію у перевірці запису у базі даних інформаційно-комунікаційної системи на цілісність та відповідність введеним користувачем даних.

Ініціалізація перемінної *transactions* для взаємодії з вузлами мережі.

Крок 3.1: Генерація значення консенсусу *proof* за обраним алгоритмом Proof-of-Work. Даний параметр буде відправлено на вузли мережі для обчислення, і являється доказом виконаної роботи.

Крок 3.2: Отримання хеш-суми попереднього блоку *previous_hash*. Вказання даного параметру створює послідовність блоків – блокчейну, а застосування даного параметру гарантує незмінність блоків, і як наслідок даних, введених у інформаційно-комунікаційну систему.

Крок 4: Вузол мережі блокчейну отримує транзакцію, зчитуючи структуру блоку і його параметри.

Крок 4.1: Вузол мережі блокчейну ініціалізує застосування алгоритму консенсусу Proof-of-Work, і після необхідних криптографічних перетворень отримує значення параметру блоку *proof* і оновлює даний параметр.

Крок 4.2: Вузол мережі блокчейну, на якому було виконано обчислення блоку зберігає локальну копію блоку, до якого, за необхідністю, можливо звернутись для перевірки цілісності.

Крок 4.3: Вузол мережі блокчейну відправляє оновлений блок на сервер.

Крок 5: Серверна частина модулю захисту отримує результати обчислень блоків вузлами мережі.

Крок 5.1: При отриманні блоку серверна чатсина перевіряє параметр індексу блоку *index*.

У разі, якщо значення параметру *index* відповідає значенню довжині блокчейну, або є меншим, то блок вважається не актуальним і не вноситься у блокчейн, виконується повернення до кроку 5.

Якщо значення параметру *index* більший за довжину блокчейну на одиницю, то даний блок вважається актуальним, виконується перехід до кроку 5.2.

Крок 5.2: Серверна частина модулю зчитує значення парметру *proof*, та обчислює його правильність.

Якщо отриманий результат підтверджує правильність виконаної роботи виконується перехід до кроку 6.

Якщо отриманий результат не є таковим, що підтверджує правильність виконаної роботи вузлом мережі блокчейну, то даний блок відкидається, виконується повернення до кроку 5.

Крок 6: Отриманий блок отримує статус підтвердженим за обраним алгоритмом консенсусу Proof-of-Work та алгоритмом вирішення проблеми сумісності результатів обчислень блоків мережі блокчейну.

Крок 6.1: Серверна частина модулю захисту вносить блок до блокчейну, тим самим збільшує його довжину на 1.

Крок 6.2: Серверна частина модулю захисту записує блокчейн у базу даних модулю захисту.

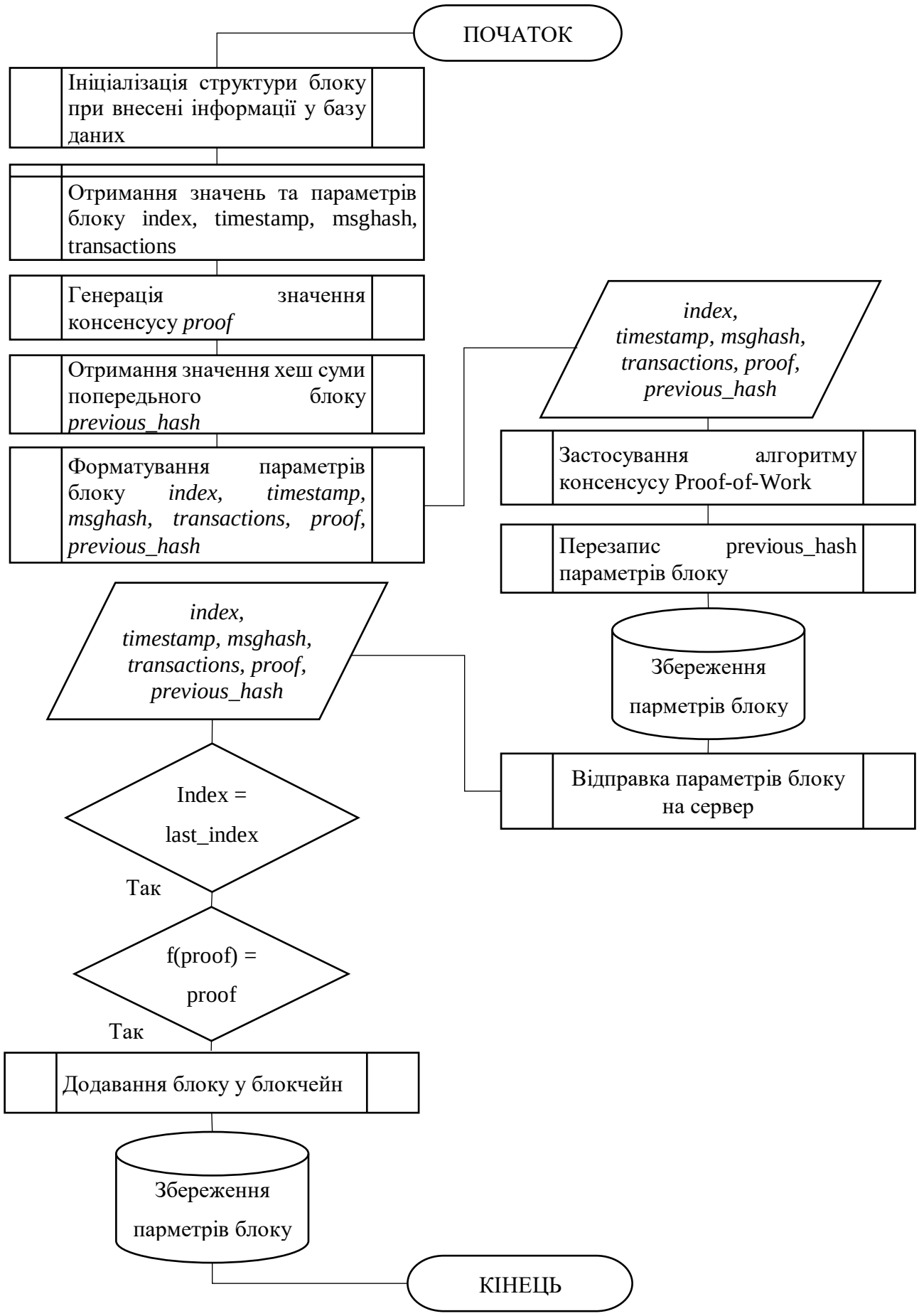

Рисунок 2.2 – Узагальнений алгоритм модулю захисту

2.3 Порівняльний аналіз розробленої підсистеми захисту бази даних

Використовуючи розроблений алгоритм підсистеми захисту бази даних та аналіз існуючих систем та засобів захисту баз даних виконаємо порівняльний аналіз щодо наведених атак (Рисунок 2.3)

Система захисту	Неправомірна модифікація осіб з безпосереднім доступом до бази даних	Шифрування пристроїв	SQL ін'єкція	Переповнення буферу обміну	Несанкціонований доступ до бази даних з зовнішнього середовища	Несанкціонована модифікація
Acra-database-protection-suite					✓	
Imperva SecureSphere Data Security					✓	
McAfee Vulnerability Manager for Databases			✓	✓	✓	
Розроблена підсистема захисту	✓	✓			✓	✓

Рисунок 2.3 – Порівняльний аналіз розробленої підсистеми захисту

2.4 Обґрунтування вибору мови програмування та засобів реалізації.

Виходячи з особливостей розробленого алгоритму роботи модулю захисту бази даних є доцільним застосувати мову програму Python та технологію API для взаємодії з інформаційно-комунікаційною системою [12]. Реалізація даного рішеня допоможе максимально ефективно та швидко використати розроблений модуль захисту бази даних, та забезпечить максимальну інтеграцію до інформаційно-комунікаційної системи.

Python — інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Розроблена в 1990 році Гвідо

ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована.

Серед основних її переваг можна назвати такі:

- чистий синтаксис (для виділення блоків слід використовувати відступи);
- переносність програм (що властиве більшості інтерпретованих мов);
- стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);
- можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);
- стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане мовою Python;
- зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);
- відкритий код (можливість редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

В якості прикладу інформаційно-комунікаційної системи було використано фреймворк Flask. Для зберігання даних обрано базу даних SQLite.

SQLite — полегшена реляційна система керування базами даних. Втілена у вигляді бібліотеки, де реалізовано багато зі стандарту SQL-92. Сирцевий код SQLite поширюється як суспільне надбання, тобто може використовуватися без обмежень та безоплатно з будь-якою метою [13]. Фінансову підтримку розробників SQLite здійснює спеціально створений консорціум, до якого входять такі компанії, як Adobe, Oracle, Mozilla, Nokia, Bentley і Bloomberg.

З 2018 року SQLite, як й JSON та CSV, рекомендований Бібліотекою Конгресу США формат зберігання структурованого набору даних.

У 2005 році проєкт отримав нагороду Google-O'Reilly Open Source Awards.

Особливістю SQLite є те, що вона не використовує парадигму клієнт-сервер, тобто рушій SQLite не є окремим процесом, з яким взаємодіє застосунок, а надає бібліотеку, з якою програма компілюється і рушій стає складовою частиною програми. Таким чином, як протокол обміну використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому комп'ютері, на якому виконується застосунок. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції весь файл, що зберігає базу даних, блокується [14]; ACID-функції досягаються зокрема за рахунок створення файлу-журналу.

Кілька процесів або потоків можуть одночасно без жодних проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, коли жодних інших запитів у цей час не обслуговується; інакше спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

2.5 Висновки до другого розділу

У другому розділі дипломної роботи було виконано вдосконалення системи захисту баз даних від несанкціонованої модифікації шляхом застосування технологій блокчейн та PoW, розроблено алгоритм роботи та наведено порівняльний аналіз розробленого модулю відносно аналогів. Виконано обґрунтування вибору мови програмування та засобів реалізації

3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ РОЗРОБЛЕНОЇ СИСТЕМИ ЗАХИСТУ

3.1 Програмна реалізація блокчейну

Блокчейн – це захищений від несанкціонованого доступу цифровий реєстр загального користування, який веде облік транзакцій у публічній або закритій одноранговій мережі. Розподілений між усіма вузлами мережі реєстр безперервно записує історію операцій з файлами між одноранговими (одного порядку) вузлами мережі як блоків інформації. Усі затверджені блоки транзакцій з'єднуються в ланцюжок - з початкового блоку до останнього доданого, звідси і назва технології - блокчейн (англ. block chain - ланцюжок блоків). Таким чином, застосування блокчейну забезпечить підвищення рівня захищеності бази даних доказової інформаційної системи за рахунок створення децентралізованого реєстру взаємодій з базою даних, інформація якого зберігається як на сервері, так і на вузлах мережі блокчейну.

Для програмної реалізації блокчейну створимо класс Blockchain[15]. Її початкові інструменти реалізовано на рисунку 1:

```
class Blockchain:
    def __init__(self):
        self.current_transactions = []
        self.chain = []
        self.nodes = set()
        # Генезис:
        self.new_block(previous_hash='1', proof=100)

    def register_node(self, address):
        parsed_url = urlparse(address)
        if parsed_url.netloc:
            self.nodes.add(parsed_url.netloc)
        elif parsed_url.path:
            self.nodes.add(parsed_url.path)
        else:
            raise ValueError('URL не є валідним')
```

Рисунок 3.1 – Реалізація початкових інструментів класу блокчейн

Для підтримки мережі блокчейну реалізуємо функцію обробки блоку на вузлах мережі (Рисунок 2):

```
def mine():
    # Використовуємо функцію PoW для останнього блоку:
    last_block = blockchain.last_block
    proof = blockchain.proof_of_work(last_block)
    blockchain.new_transaction(
        sender="0",
        recipient=node_identifier,
        amount=1,
    )
    # Додаємо блок до блокчейну
    previous_hash = blockchain.hash(last_block)
    block = blockchain.new_block(proof, previous_hash)
    response = {
        'message': "Новий блок оброблено",
        'index': block['index'],
        'transactions': block['transactions'],
        'proof': block['proof'],
        'previous_hash': block['previous_hash'],
    }
    return jsonify(response), 200
```

Рисунок 3.2 – Ініціалізація блоку

Оскільки в розробленій системі модулю захисту бази даних може застосовуватись декілька узлів підтримки блокчейну, то виникає необхідність реалізувати функцію, застосування якої вирішує проблему конфліктності вузлів [16] щодо послідовності блоків у блокчейну, оскільки швидкість обробки алгоритму консенсусу може змінюватись в залежності від апаратного забезпечення кожного вузла мережі (Рисунок 7):

```python
def resolve_conflicts(self):

    neighbours = self.nodes
    new_chain = None

    #Отримаємо довжину блокчейну
    max_length = len(self.chain)

    # Отримаємо довжини всіх блокчейну з кожного із вузлів мережі
    for node in neighbours:
        response = requests.get(f'http://{node}/chain')

        if response.status_code == 200:
            length = response.json()['length']
            chain = response.json()['chain']

            # Перевірка найдовшого блокчейну з отриманих результатів
            if length > max_length and self.valid_chain(chain):
                max_length = length
                new_chain = chain

    # Заміна блокчейну при отриманні нового результату
    if new_chain:
        self.chain = new_chain
        return True

    return False
```

Рисунок 3.3 – функція вирішення проблеми сумісності вузлів мережі

3.2 Програмна реалізація алгоритму Proof-of-Work

Замість того, щоб звертатися до третіх осіб, наприклад, як до посередників при проведенні транзакцій, вузли блокчейн-мережі використовують спеціальний протокол консенсусу для узгодження вмісту реєстру, а також криптографічні алгоритми хешування та електронно-цифрові підписи для забезпечення цілісності транзакції передачі параметрів.

Механізм консенсусу гарантує, що розподілені реєстри є точними копіями, що знижує ризик зміни інформації і базі даних доказової інформаційно-

комунікаційної системи і забезпечить підвищення захищеності від несанкціонованої модифікації. Криптографічні алгоритми хешування, такі як алгоритм обчислень SHA256 [17], гарантують, що будь-яка зміна вхідних даних транзакції, навіть незначна, призведе до появи іншого значення хешу в результатах розрахунків, що вказує на ймовірність компрометації вхідних даних транзакції. Електронно-цифрові підписи гарантують, що транзакції здійснюються легітимними відправниками (підписані закритими ключами), а чи не зловмисниками. Децентралізована однорангова блокчейн-мережа позбавляє окремих учасників чи груп учасників можливості контролювати базову інфраструктуру чи дестабілізувати всю систему. Усі учасники мережі рівні та підключаються до неї за одними й тими же протоколами. Учасниками може бути фізичні особи, державні структури, організації чи об'єднання всіх перелічених типів учасників. По суті, система записує хронологічний порядок проведення транзакцій з усіма вузлами мережі, що визнали дійсність транзакцій за допомогою обраної моделі консенсусу. Результатом є транзакції, що не підлягають відміні, узгоджені всіма учасниками мережі децентралізовано.

Для програмної реалізації створемо функції за обраним алгоритмом консенсусу, та функцію перевірки останньої (Рисунок 4)

```
class Blockchain(object):
    ...

    def proof_of_work(self, last_proof):
        proof = 0
        while self.valid_proof(last_proof, proof) is False:
            proof += 1

        return proof

    @staticmethod
    def valid_proof(last_proof, proof):

        guess = f'{last_proof}{proof}'.encode()
        guess_hash = hashlib.sha256(guess).hexdigest()
        return guess_hash[:4] == "0000"
```

Рисунок 3.4 – реалізовані функції як інструменти класу Blockchain

Щоб скоригувати складність алгоритму, ми можемо змінити кількість нулів . У нашому випадку, 4 – достатньо. Внесення одного нуля створює значну різницю у часі, необхідному для пошуку рішення за обраним алгоритмом консенсусу для обробки транзакції.

3.3 Створеня серверного додатку модулю захисту доказової бази даних

Для реалізації серверного додатку модулю захисту доказової бази даних було обрано фреймворк Flask. Також для створення блокчейну необхідно застосувати часові мітки, реалізація яких покладена на бібліотеку time. Дана бібліотека преставляє час з точністю до секунди, а зручний формат даних юнікс дозволяє легко зберігати та в необхідності швидко конвертувати часову мітку у звичний формат.

Відображення блокчейну буде відбуватись за допомогою формату даних json, який в зручному форматі відображає параметри блоків та їх послідовність. Функція створення нової транзакції блоку, який в подальшому буде добавлений до блокчейну (Рисунок 5):

```
def new_transaction():
    values = request.get_json()
    # Перевірка на наявність всіх параметрів функції при запиті до серверу
    required = ['sender', 'recipient', 'amount']
    if not all(k in values for k in required):
        return 'Missing values', 400

    # Створення нової транзакції
    index = blockchain.new_transaction(values['sender'], values['recipient'], values['amount'])
    response = {'message': f'Транзакція буде добавлена до блоку {index}'}
    return jsonify(response), 201
```

Рисунок 3.5 – Реалізована функція створення нової транзакції

Після обробки транзакції та внесення блоку до блокчейну потрібно відобразити зміни, оновивши параметри системи. Реалізуємо даний функціонал шляхом перезапуску значень параметрів та створення нового блоку після обробки попереднього, використовуючи формат даних json (Рисунок 6):

```
def new_block(self, proof, previous_hash):

    # Внесення даних у форматі json

    block = {
        'index': len(self.chain) + 1,
        'timestamp': time(),
        'DBfileindex': self.dbindex,
        'transactions': self.current_transactions,
        'proof': proof,
        'previous_hash': previous_hash or self.hash(self.chain[-1]),
    }

    # Перезапис транзакції для обробки вузлами мережі

    self.current_transactions = []

    # Додавання блоку до загального списку блокчейну
    self.chain.append(block)
    return block
```

Рисунок 3.6 – Функція створення нового блоку

Щоб відображити повну послідовність блокчейну та його складових (блоків) використаємо json формат. Для цього реалізуємо відповідну функцію (Рисунок 6):

```
@app.route('/chain', methods=['GET'])
def full_chain():
    response = {
        'chain': blockchain.chain,
        'length': len(blockchain.chain),
    }
    return jsonify(response), 200
```

Рисунок 3.7 – Відображення послідовності блоків всього блокчейну

3.4 Тестування роботи розробленої системи захисту

Для виконання тестування розробленого модулю підвищення рівня захищеності бази даних доказової інформаційно-комунікацйної системи від несанкціонованої модифікації на основі технологій блокчейн та PoW використаємо бібліотку для автоматичного тестування unittest та концепіцію test case.

Test case – блок тестування. Він валідує відповіді на різні набори вхідних даних. Модуль unittest надає базовий клас TestCase, який застосовується для створення нових тестових ітерацій даних, які будуть надіслані функціям для перевірки варіативності і рівня обробки вхідних даних функцій і можливих помилок при роботі модулю захисту [18].

Для виконання тестування імпортуємо необхідні бібліотеки, а саме:

– hashlib – необхідна для виконання тестів функцій з криптографічними перетвореннями, а саме функцій застосування та перевірки алгоритму Proof-of-Work;
– json – застосовується для взаємодії з функціями, які отримують та надсилають данні у форматі json
– unittest – бібліотека тестування, на основі якої виконується перевірка програмного коду модулю захисту
– blockchain – ядро розробленого модулю захисту, який відповідає за криптографічні перетворення у алгоритмі роботи Proof-of-Work, ініціалізацію блоків, управлінням блокчейну та його перевіркою, створенням транзакцій.

Наступним кроком створемо необхідні функції для перевірки розробленого модулю, після чого отримаємо загальний результат та час, витрачений на проведення тестування.

Для перевірки створення нової транзакції створемо наступний тест (Рисунок 3.8):

```
    def test_reset_transaction(self):
        self.create_transaction()

        initial_length = len(self.blockchain.current_transactions)
        self.create_block()
        current_length = len(self.blockchain.current_transactions)
        assert initial_length == 1
        assert current_length == 0
```

Рисунок 3.8 – Функція

Для перевірки серверного додатку модулю захисту на створення та перевірку значення алгоритму Proof-of-Work застосуємо наступний тест (Рисунок 3.9):

```
class CheckPoW(BlockchainTest):

    def checkhash(self):
        self.create_block()
        new_block = self.blockchain.last_block
        new_block_json = json.dumps(self.blockchain.last_block, sort_keys=True).encode()
        new_hash = hashlib.sha256(new_block_json).hexdigest()

        assert len(new_hash) == 64
        assert new_hash == self.blockchain.hash(new_block)
```

Рисунок 3.9 – Тест для перевірки помилок серверного додатку при застосуванні алгоритму PoW

Для перевірки функції отримання останнього блоку блокчейну використаємо наступний тест (Рисунок 3.10):

```
def test_get_last_block(self):
    self.create_block()
    created_block = self.blockchain.last_block

    assert len(self.blockchain.chain) == 2
    assert created_block is self.blockchain.chain[-1]
```

Рисунок 3.10 – Тест для перевірки функції повернення параметрів останнього блоку

Виконаємо тестування реєстрації вузла та його відображення у загальному списку доступних вузлів. Для цього застосуємо наступний тест (Рисунок 3.11):

```
class Test_Register_Nodes(BlockchainTest):

    def test_check_node_in_pool(self):
        blockchain = Blockchain()

        blockchain.register_node('http://192.168.0.1:6500')

        self.assertIn('192.168.0.1:6500', blockchain.nodes)
```

Рисунок 3.11 – Тестування створення та перевірки вузла на відображення

Отриманий результат тестування свідчить про достатній рівень обробки помилок реалізованих функцій модулю захисту (Рисунок 3.12):

```
Process finished with exit code 0
Test 4 successfully passed
---------------
Test 3 successfully passed


Ran 4 tests in 5.014s

OK
---------------
Test 1 successfully passed
---------------
Test 2 successfully passed
---------------
```

Рисунок 3.12 – Пройдене тестування обраних функцій модулю захисту

3.5 Висновки до третього розділу

У третьому розділі дипломної роботи було виконано програмну реалізацію блокчейну та алгоритму консенсусу “proof-of-work”, на основі яких створено та протестовано серверну та клієнтську частину модулю захисту.

4 ЕКОНОМІЧНА ЧАСТИНА

Мета даного розділу дипломної роботи полягає у обгрунтуванню економічної доцільності реалізації захисту доказової бази даних інформаційно-комунікаційної системи за допомогою технологій блокчейн та PoW.

Для досягання вищенаведеної мети необхідно:

- Виконати технологічний аудит розробки
- Оцінити прогнозування витрат на виконання дослідної та конструкторсько-технологічної роботи
- Виконати прогнозування комерційних ефектів від реалізації результатів розробки
- Розрахувати ефективність вкладених інвестицій та період їх окупності

4.1 Оцінювання комерційного потенціалу розробки (технологічний аудит розробки)

Для оцінки комерційного потенціалу розробки було залучено трьох незалежних експертів: Експерт1, Експерт2, Експерт3. Кожен з експертів повинен ознайомитись з виконаною розробкою для підвищення захищеності доказової бази даних інформаційно-комунікаційної системи та внести відповідні оцінки по п’ятибальній системі щодо заданих критеріїв. Їх середньоарифметична сума балів являє собою оцінку незалежної експертної групи, яка і буде означати рівень комерційного потенціалу розробки. Критерії та оцінки експертів наведено в таблиці 4.1:

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та варіанти оцінювання

Критерії	Прізвище, ініціали експерта		
	Експерт 1	Експерт 2	Експерт 3
	Бали, виставлені експертами:		
Технічна здійсненність концепції	4	4	5

Кількість аналогів на ринку	4	5	4
Ціна продукту відносно аналогів	4	5	5
Властивості продукту відносно аналогів	4	4	3
Експлуатаційні витрати відносно аналогів	3	4	4
Ринкова динаміка	4	4	4
Конкуренція на ринку	3	3	2
Кількість фахівців для підтримки продукту	3	4	4
Термін реалізації ідеї, окупність інвестицій	4	3	4
Необхідна кількість регламентованих документів на виробництво та реалізацію продукту	3	3	4
Сума балів	36	39	39
Середньоарифметична сума балів	$\overline{СБ} = \frac{\sum_{1}^{i} СБ_i}{i} = \frac{x1 + x2 + x3}{3} = 38$		

Аналізуючи результат оцінювання експертної групи, наведений в таблиці 5.1, можна зробити висновок щодо рівня комерційного потенціалу розробки модулю захисту бази даних доказової інформаційно-комунікаційної системи. Використаємо таблицю відповідностей середнього балу оцінювання незалежної експертної та рівнів комерційного потенціалу розробки модулю (Таблиця 4.2):

Середньоарифметична сума балів	Рівень комерційного потенціалу
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-50	Високий

Отже, відповідно до отриманих результатів оцінювання експернтої групи рівня комерційного потенціалу розробки модулю захисту доказової бази даних інформаційно-комунікацйної системи отримаємо результат: Вище середнього.

Оцінювання рівня якості розробки модулю захисту доказової бази даних здійснюється з метою порівняльного аналізу та визначення найбільш ефективного, з технічної точки зору, варіанта інженерного рішення.

Рівень якості – кількісна характеристика міри придатності певного виду продукції для задоволення конкретного попиту на неї при порівнянні з відповідними базовими показниками за фіксованих умов споживання.

Абсолютний рівень якості розробки модулю захисту доказової бази даних знаходимо обчисленням обраних для вимірювання показників, не порівнюючи їх із відповідними показниками аналогічних засобів.

Для визначення рівня якості розробки обрано систему параметрів: ймовірність автентифікації порушника, швидкість автентифікації, об’єм даних, що передаються, кількість взаємодій між сторонами.

Визначаємо величину параметрів якості в балах та встановлюємо граничні значення (кращі, гірші, середні). Дані для кожного параметра представлено у таблиці 4.3.

Таблиця 4.3 – Основні параметри модулю захисту бази даних доказової інформаційно-комунікаційної системи

Параметри	Абсолютне значення параметра			Коефіцієнт вагомості параметра
	Краще +5…+4	Середнє +3	Гірше +1..+2	
ймовірність автентифікації порушника	4			0.6
швидкість автентифікації			2	0.2
об’єм даних ,що передаються	4			0.1

кількість взаємодій			2	0.1

Із врахуванням коефіцієнтів вагомості відповідних параметрів можна визначити абсолютний рівень якості інноваційного рішення за формулою

$$K_{я.а.} = \sum_{i=1}^{n} Рн_i * a_i = 4 \cdot 0{,}6 + 2 \cdot 0{,}2 + 4 \cdot 0{,}1 + 2 \cdot 0{,}1 = 3{,}4 \qquad (5.1)$$

де $Рн_i$ – числове значення i-го параметра інноваційного рішення, n – кількість параметрів інноваційного рішення, що прийняті для оцінювання, a_i – коефіцієнт вагомості відповідного параметра (сума коефіцієнтів вагомості всіх параметрів повинна дорівнювати 1).

Отже, абсолютний рівень якості підвищення рівня захищеності бази даних доказової інформаційно-комунікаційної системи від несанкціонованої модифікації складає 3,4 бали.

4.2 Прогнозування витрат на виконання дослідної та конструкторсько-технологічної роботи

Команда розробки методу та засобу автентифікації користувачів з нульовим знанням складається з адміністратора (керівника), інженер-програміста та тестувальника.

Основна заробітна плата для команди розробників *Зо*, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_О = \frac{M}{T_p} t \qquad (4.2)$$

де M – місячний посадовий оклад розробника;

T_p – кількість робочих днів у місяці, T_p = 22 дні; t – число днів роботи.

Розрахунки заробітної плати для розробників наведені в таблиці 4.4

Заробітна плата наукового керівника:

$$З_{О_{НК}} = \frac{23000}{22} = \times 9 = 9409{,}09 \text{ (грн)}$$

Заробітна плата розробника:

$$З_{О_Р} = \frac{21000}{22} = \times 22 = 21000 \text{ (грн)}$$

Заробітна плата тестувальника:

$$З_{О_т} = \frac{18000}{22} = \times 7 = 5727{,}27 \text{ (грн)}$$

Витрати на оплату праці, основна заробітна плата:

$$З_О = З_{О_{НК}} + З_{О_Р} = 9409{,}09 + 21000 + 5727{,}27 = 36136{,}36$$

Таблиця 4.4 – Розрахунки основної заробітної плати розробників

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Керівник	23000	1045,45	9	9 409,09
Розробник	21000	954,54	22	21000
Тестувальник	18000	818,18	7	5727,27
Всього:				36136,36

Основна заробітна плата робітників *Зр*, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$З_Р = \sum_{i=1}^{n} t_i \cdot C_i \,, \qquad (5.6)$$

де t_i – норма часу (трудомісткість) на виконання конкретної роботи, годин; n – число робіт по видах та розрядах; C_i – погодинна тарифна ставка робітника відповідного розряду, який виконує дану роботу. C_i визначається за формулою:

$$C_i = \frac{M_M * K_i}{T_p * T_{ЗМ}}, \qquad (5.7)$$

де $М_М$ – розмір мінімальної заробітної плати за місяць, грн.; в 2021 році мінімальна заробітна плата становить – 6500 грн., $К_і$ – тарифний коефіцієнт робітника відповідного розряду, $Т_р$ = 22 дні; $Т_{зм}$ – тривалість зміни, $Т_{зм}$ =8 годин.

Таблиця 4.5 – Заробітна плата робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Розробка	8	5	56,8	1,54	454,5
Тестування	8	4	53.6	1,45	428,8
Впровадження	2	2	40,1	1,09	80,2
Всього					963,2

Додаткова заробітна плата $З_Д$ всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_Д = 0.1*(З_О + З_Р) = 0{,}1*(36{,}136{,}36 + 963{,}2) = 3709{,}98 \text{ (грн.)} \quad (5.8)$$

Нарахування на заробітну плату $Н_{ЗП}$ розраховується як 22% від суми основної та додаткової заробітної плати:

$$Н_{ЗП} = (З_О + З_Д) \times \frac{\beta}{100} = (36136{,}36 + 963{,}2 + 3709{,}98) \cdot 0{,}22 = 8978{,}085 \text{ грн} \quad (5.9)$$

де ЗО – основна заробітна плата розробників, грн.;

Зр – основна заробітна плата робітників, грн.;

ЗД – додаткова заробітна плата розробників, грн.;

β – ставка єдиного внеску на загальнообов’язкове державне страхування.

Розрахунок амортизаційних відрахувань виконується за такою формулою:

$$А = \frac{Ц \times Н_А}{100} \times \frac{Т}{12} \quad (5.10)$$

де $Ц$ – балансова вартість обладнання, грн;

H_A – річна норма амортизаційних відрахувань. Для даного випадку можна прийняти, що H_A = 25%;

T – термін використання (T=6 місяців);

T_B – корисний час використання (T_B для комп'ютера становить 4 роки).

Отже, розрахуємо амортизаційні відрахування:

$$A = \frac{25000 \times 25}{100} \times \frac{6}{12} = \frac{625000}{100} \times 0{,}5 = 6250 \times 0{,}5 = 3125 \text{ (грн)}$$

Під час виконання розробки використовувався ноутбук вартістю 25000 грн. Амортизаційні відрахування для ноутбуку представлені у таблиці 4.6

Таблиця 4.6 - Амортизаційні відрахування

Найменування	Ціна, грн.	Норма амортизації, %	Термін використання, міс.	Сума амортизації, грн.
Ноутбук	25000	25	6	6375
Всього	3125			

Витрати на силову електроенергію розраховуються за формулою:

$$B_E = B \times П \times Ф \times K_П \qquad (5.11)$$

де В – вартість 1кВт-години електроенергії (В=2,44 грн/кВт);

П – установлена потужність комп'ютеру (П=0,74 кВт);

Ф – фактична кількість годин роботи комп'ютеру (Ф=200 год);

$K_П$ – коефіцієнт використання потужності (Кп<1, Кп = 0,8).

Відповідно до формули 5.11 витрати на силову електроенергію:

$$B_E = 2{,}44 \times 0{,}74 \times 200 \times 0{,}8 = 288{,}9 \text{ (грн.)}$$

Інші витрати $B_{ін}$ можна прийняти як (100-300)% від суми основної заробітної плати розробників, які виконували роботу, тобто:

$$B_{ін} = 1*(36136{,}3 + 963{,}2) = 37099{,}5 \text{ (грн.)} \qquad (5.11)$$

Сума усіх попередніх витрат дає загальні витрати на виконання роботи. Усі витрати складають:

В = 36136,3 + 963,2 + 3709,98 + 8978,085 + 37099,5 = 86887,065 (грн.)

Розрахуємо загальну вартість наукової розробки $В_{заг}$ за формулою:

$$B_{заг} = \frac{B}{\alpha}, \tag{5.12}$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відносних одиницях.

$B_{заг} = \frac{B}{\alpha}$ = 86887,065 /1 = 86887,065 (грн)

Прогнозування загальних витрат *ЗВ* на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{B_{заг}}{\beta} \tag{5.13}$$

Розрахунок прогнозованих загальних витрат:

ЗВ = 86887,065 / 0,7 = 124 124,379 (грн.)

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

В економічному розділі магістерської кваліфікаційної роботи обґрунтовується економічна доцільність розробки підвищення захищеності баз даних доказових інформаційно-комунікаційних систем за допомогою технологій блокчейн та PoW. На виконання усіх необхідних робіт необхідно 40 робочих днів.

Зростання чистого продукту для даного засобу можна оцінити у теперішній вартості грошей. Зростання чистого прибутку забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності.

Збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_{1}^{n} (\Delta П_Я \times N + П_Я \Delta N)_i$$

де $\Delta П_Я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_Я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

В результаті впровадження результатів наукової розробки модулю захисту витрати на розробку алгоритму зменшаться на 4000 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 4000 грн), а кількість користувачів збільшиться: протягом першого року – на 5 користувачів, протягом другого року – на 6 користувачів, протягом третього року – на 8 користувачів.

Реалізація продукції до впровадження результатів наукової розробки складала 16 шт., а прибуток, що його отримувало підприємство (організація) на одиницю продукції до впровадження результатів наукової розробки – 10000 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Отже, збільшення чистого продукту $\Delta П_1$ протягом першого року складатиме:

$$\Delta П_1 = 1000 \times 16 + (4000 + 10000) \times 5 = 230000 \text{ (грн)}$$

Протягом другого року:

$$\Delta П_2 = 10000 \times 16 + (4000 + 14000) \times (5 + 6) = 358000 \text{ (грн)}$$

Протягом третього року:

$$\Delta П_3 = 10000 \times 16 + (4000 + 14000) \times (5 + 6 + 8) = 502000 \text{ (грн)}$$

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Розрахунок ефективності вкладених інвестицій передбачає проведення таких робіт:

1-й крок. Розрахуємо теперішню вартість інвестицій PV, що вкладаються в наукову розробку. Такою вартістю ми можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, розраховану раніше, тобто будемо вважати, що ЗВ = PV = 124 124,379 грн.

2-й крок. Розрахуємо очікуване збільшення прибутку $\Delta П_i$, що його отримає підприємство (організація) від впровадження результатів наукової розробки, для кожного із років, починаючи з першого року впровадження. Таке збільшення прибутку також було розраховане нами раніше та становить:

$$\Delta П_1 = 230000 \text{ грн}, \Delta П_2 = 358000 \text{ грн}, \Delta П_3 = 502000 \text{ грн}$$

3-й крок. Для спрощення подальших розрахунків необхідно побудувати вісь часу, на яку наносять всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів.

Якщо загальні витрати ЗВ на виконання та впровадження результатів НДДКР (або теперішня вартість інвестицій PV) дорівнюють 124 124,379 грн., а результати вкладених у наукову розробку інвестицій почнуть виявлятися вже в кінці другого року впровадження. То ці результати виявляться у тому, що у першому році підприємство отримає збільшення чистого прибутку на 230000 грн. відносно базового року, у другому році – збільшення чистого прибутку на 358000 грн (відносно базового року), у третьому році – збільшення чистого прибутку на 502000 грн (відносно базового року).

4-й крок. Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$. Для цього скористаємося формулою:

$$E_{абс} = (ПП - PV)$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство (організація) від реалізації результатів наукової розробки, грн.;

PV – теперішня вартість інвестицій PV = ЗВ = 124 124,379 грн.

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)}$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

Т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні. Для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки «0».

$\Delta П_1 = 230000$ грн, $\Delta П_2 = 358000$ грн, $\Delta П_3 = 502000$ грн

$$ПП = \frac{230000}{(1+0{,}1)^2} + \frac{358000}{(1+0{,}1)^3} + \frac{502000}{(1+0{,}1)^3} = \frac{230000}{1{,}21} + \frac{358000}{1{,}331} + \frac{502000}{1{,}4641} =$$
$$190082{,}64 + 268970{,}7 + 343600{,}27 = 802653{,}6\ (грн)$$

$$E_{абс} = 802653{,}6 - 124\,124{,}379 = 678529{,}232\ \text{(грн)}$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

5-й крок. Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій E_B. Для цього використаємо формулу:

$$E_B = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1,$$

де $Е_{абс}$ – абсолютна ефективність вкладених інвестицій, грн; PV – теперішня вартість інвестицій PV = ЗВ, грн; $Т_ж$ – життєвий цикл наукової розробки, роки.

Далі, розрахована величина $Е_в$ порівнюється з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau = d + f$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2021 році в Україні d = (0,14...0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1), але можу бути і значно більше.

Якщо величина $E_B > \tau_{мін}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки. В іншому випадку фінансування наукової розробки здійснюватися не буде.

Спочатку спрогнозуємо величину $\tau_{мін}$. Припустимо, що за даних умов

$\tau_{мін} = 0{,}15 + 0{,}05 = 0{,}2$.

Тоді відносна (щорічна) ефективність вкладених інвестицій в проведення наукових досліджень та впровадження їх результатів складе:

$$E_B = \sqrt[4]{1 + \frac{802653{,}6}{124124{,}379}} - 1 = \sqrt[4]{1 + 6{,}46} - 1 = \sqrt[4]{7{,}46} - 1 = 1.65 - 1 = 0{,}65$$ або 65%.

Оскільки $E_B = 26\% > \tau_{мін} = 0{,}2 = 20\%$, то інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

6-й крок. Розраховують термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $Т_{ок}$ можна розрахувати за формулою:

$$T_{OK} = \frac{1}{E_B}$$

Якщо $T_{ОК}$<3...5-ти років, то фінансування даної наукової розробки в принципі є доцільним. В інших випадках потрібні додаткові розрахунки та обґрунтування.

$$T_{ОК} = \frac{1}{0,65} = 1,53 \text{ року}$$

$T_{ОК}$<5 років, що свідчить про доцільність фінансування даної наукової розробки.

4.5 Висновки до розділу

Рівень комерційного потенціалу модулю захисту доказової бази даних інформаційно-комунікаційної системи з застосуванням технологій блокчейн та PoW є вище середнього. Загальні витрати на створення нового програмного продукту склали 124 124,379 грн. Абсолютна ефективність капіталовкладень для даної розробки за 3 роки 802653,6 грн. Показники ефективності показують, що дана розробка є доцільною. Термін окупності розробленого проекту менше 5-ти років, що підтверджує доцільність вкладання коштів в дану розробку.

ВИСНОВКИ

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 1. Acra єдине рішення для захисту життєвого циклу даних. [Електронний ресурс] - Режим доступу: https://kz.iitd.com.ua/wp-content/uploads/2020/03/acra-feature-presentation-q3-ua-iitd.pdf (дата звернення: 21.09.2021)

2. 2. Imperva SecureSphere Data Security. [Електронний ресурс] - Режим доступу: https://www.imperva.com/resources/datasheets/DS_SecureSphere_Data-Security.pdf (дата звернення: 23.09.2021)

3. 3. McAfee Vulnerability Manager for Databases. [Електронний ресурс] - Режим доступу: https://www.mcafee.com/enterprise/ru-ru/assets/data-sheets/ds-vulnerability-manager-for-databases.pdf (дата звернення: 24.09.2021)

4. 4. Consensus - Immutable agreement for the Internet of value. [Електронний ресурс] - Режим доступу: https://assets.kpmg/content/dam/kpmg/pdf/2016/06/kpmg-blockchain-consensus-mechanism.pdf (дата звернення: 01.10.2021)

5. 5. What Is a 51% Attack? [Електронний ресурс] - Режим доступу: https://www.coindesk.com/learn/what-is-a-51-attack/ (дата звернення: 04.10.2021)

6. 6. Research and Challenges on Bitcoin Anonymity. [Електронний ресурс] – Режим доступу: http://www.indiaenvironmentportal.org.in/files/file/Bitcoin%20 Anonymity.pdf (дата звернення: 09.10.2021)

7. Біткоїн. [Електронний ресурс] – Режим доступу: - https://uk.wikipedia.org/wiki/%D0%91%D1%96%D1%82%D0%BA%D0%BE%D1%97%D0%BD (дата звернення: 12.10.2021)

8. What is a smart contract? [Електронний ресурс] – Режим доступу: https://capital.com/smart-contracts-definition (дата звернення: 17.10.2021)

9. - Pavan Duggal Blockchain Contracts and Cyberlaw / Pavan Duggal – K. : Information Systems (дата звернення: 24.10.2021)

10. What is Gas? [Електронний ресурс] – Режим доступу: https://kb.myetherwallet.com/en/transactions/what-is-gas/ (дата звернення: 26.10.2021)

11. The proof of work vs proof of stake: an in-depth discussion. [Електронний ресурс] – Режим доступу: https://www.leewayhertz.com/proof-of-work-vs-proof-of-stake/ (дата звернення: 01.11.2021)

12. Майк МакГрат. Программирование на Python для начинающих: [перевод с англ. М.А. Райтмана] - М .: Эскмо, 2015. - 192 с

13. SQLite with Blockchain. [Електронний ресурс] – Режим доступу: https://github.com/aergoio/aergolite (дата звернення: 09.11.2021)

14. Let’s Make a Hash Chain in SQLite. [Електронний ресурс] – Режим доступу: https://www.viget.com/articles/lets-make-a-hash-chain-in-sqlite/ (дата звернення: 11.11.2021)

15. Learn Blockchain by Building One: A Concise Path to Understanding Cryptocurrencies 1st ed. Edition by Daniel van Flymen. – 2020. – 208 с

16. Consensus-mechanisms. [Електронний ресурс] – Режим доступу: https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/ (дата звернення: 13.11.2021)

17. Bitcoin: A Peer-to-Peer Electronic Cash System . [Електронний ресурс] – Режим доступу:https://www.ussc.gov/sites/default/files/pdf/training/annual-national-training-seminar/2018/Emerging_Tech_Bitcoin_Crypto.pdf (дата звернення: 17.11.2021)

18. Unit testing framework System. [Електронний ресурс] – Режим доступу: https://docs.python.org/3/library/unittest.html (дата звернення: 24.11.2021)

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

Додаток А. Технічне завдання

ЗАТВЕРДЖУЮ

Голова секції “Управління
інформаційною
безпекою” кафедри МБІС
д.т.н., професор

_____Ю.Є.Яремчук
“____” _____2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:

_____
_____
_____

08-42.МКР.004.00.000.ТЗ

Керівник магістерської кваліфікаційної

роботи

к.т.н., доцент

_____.

Вінниця – 2021 р.

1. Найменування та область застосування

Програмний модуль захисту баз даних від несанкціонованої модифікації. Область застосування: півищення рівня захищеності доказових інформаційно-комунікаційних систем від несанкціонованої модифікації.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № від 2021 р.

3. Мета та призначення розробки

3.1 Мета розробки: підвищення рівня захищеності баз даних інформаційно-комунікаційних систем.

3.2 Призначення: розроблений програмний модуль підвищує рівень захищеності доказових інформаційно-комунікаційних систем від несанкціонованої модифікації.

4. Джерела розробки

4.1. Майк МакГрат. Программирование на Python для начинающих: [перевод с англ. М.А. Райтмана] - М .: Эскмо, 2015. - 192 с

4.2. SQLite with Blockchain. [Електронний ресурс] – Режим доступу: https://github.com/aergoio/aergolite

4.3. Let’s Make a Hash Chain in SQLite. [Електронний ресурс] – Режим доступу: https://www.viget.com/articles/lets-make-a-hash-chain-in-sqlite/

4.4. Learn Blockchain by Building One: A Concise Path to Understanding Cryptocurrencies 1st ed. Edition by Daniel van Flymen. – 2020. – 208 с

4.5. Consensus-mechanisms. [Електронний ресурс] – Режим доступу: https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Реалізований модуль захисту не повинен вимагати спеціальних ліцензійних програмних додатків;

5.1.2 Розроблений модуль захисту повинен підвищувати рівень захищеності бази даних доказових інформаційно-комунікаційних систем.

5.2 Вимоги до надійності:

5.2.1 Розроблений модуль захисту повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Розроблений модуль захисту виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Intel Core i3–10300 3.7 ГГц подібні до них;

– оперативна пам’ять – не менше 8 Gb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до технічного захисту інформації

6.1 Необхідно забезпечити захист бази даних доказової інформаційно-комунікаційної системи від несанкціонованої модифікації

розроблюваного програмного засобу від несанкціонованого використання.

7. Техніко-економічні показники

7.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

7.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

8. Стадії та етапи розробки

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Початок	Закінчення
1	Визначення напрямку магістерської роботи, формулювання теми	02.09.2021	05.09.2021
2	Аналіз предметної області обраної теми	06.09.2021	11.09.2021
3	Апробація отриманих результатів	12.09.2021	01.10.2021
4	Розробка алгоритму роботи	02.10.2021	25.10.2021

5	Написання магістерської роботи на основі розробленої теми	26.10.2021	26.11.2021
6	Розробка економічної частини	27.11.2021	8.11.2021
7	Передзахист магістерської кваліфікаційної роботи	9.12.2021	10.12.2021
8	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	11.12.2021	14.12.2021
9	Захист магістерської кваліфікаційної роботи	20.12.2021	21.12.2021

10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;

– програмний модуль;

– презентація;

– відзив керівника роботи;

– відзив рецензента

Технічне завдання до виконання прийняв _____Іванюк Т.В.

Додаток Б. Код розробленого модулю захисту

```
import hashlib
import json
from time import time
from urllib.parse import urlparse
from uuid import uuid4

import requests
from flask import Flask, jsonify, request
class Blockchain:
    def __init__(self):
        self.dbindex = ()
        self.current_transactions = []
        self.chain = []
        self.nodes = set()
        # Генезис:
        self.new_block(previous_hash='1', proof=100)

    def register_node(self, address):
        parsed_url = urlparse(address)
        if parsed_url.netloc:
            self.nodes.add(parsed_url.netloc)
        elif parsed_url.path:
            self.nodes.add(parsed_url.path)
        else:
            raise ValueError('URL не є валідним')

    def valid_chain(self, chain):


        last_block = chain[0]
        current_index = 1

        while current_index < len(chain):
            block = chain[current_index]
            print(f'{last_block}')
            print(f'{block}')
            print("\n-----------\n")
            # Check that the hash of the block is correct
            last_block_hash = self.hash(last_block)
            if block['previous_hash'] != last_block_hash:
                return False

            # Check that the Proof of Work is correct
            if not self.valid_proof(last_block['proof'], block['proof'], last_block_hash):
                return False
```

```
            last_block = block
            current_index += 1

        return True

    def resolve_conflicts(self):

        neighbours = self.nodes
        new_chain = None

        #Отримаємо довжину блокчейну
        max_length = len(self.chain)

        # Отримаємо довжини всіх блокчейну з кожного із вузлів мережі
        for node in neighbours:
            response = requests.get(f'http://{node}/chain')

            if response.status_code == 200:
                length = response.json()['length']
                chain = response.json()['chain']

                # Перевірка найдовшого блокчейну з отриманих результатів
                if length > max_length and self.valid_chain(chain):
                    max_length = length
                    new_chain = chain

        # Заміна блокчейну при отриманні нового результату
        if new_chain:
            self.chain = new_chain
            return True

        return False

    def new_block(self, proof, previous_hash):

        # Внесення даних у форматі json

        block = {
            'index': len(self.chain) + 1,
            'timestamp': time(),
            'DBfileindex': self.dbindex,
            'transactions': self.current_transactions,
            'proof': proof,
            'previous_hash': previous_hash or self.hash(self.chain[-1]),
        }

        # Перезапис транзакції для обробки вузлами мережі
```

```
    self.current_transactions = []

    # Додавання блоку до загального списку блокчейну
    self.chain.append(block)
    return block

def new_transaction(self, sender, recipient, amount):

    self.current_transactions.append({
        'sender': sender,
        'recipient': recipient,
        'amount': amount,
    })

    return self.last_block['index'] + 1

@property
def last_block(self):
    return self.chain[-1]

@staticmethod
def hash(block):
    """
    Creates a SHA-256 hash of a Block
    :param block: Block
    """

    # We must make sure that the Dictionary is Ordered, or we'll have inconsistent hashes
    block_string = json.dumps(block, sort_keys=True).encode()
    return hashlib.sha256(block_string).hexdigest()

def proof_of_work(self, last_block):

    last_proof = last_block['proof']
    last_hash = self.hash(last_block)

    proof = 0
    while self.valid_proof(last_proof, proof, last_hash) is False:
        proof += 1

    return proof

@staticmethod
def valid_proof(last_proof, proof, last_hash):

    guess = f'{last_proof}{proof}{last_hash}'.encode()
    guess_hash = hashlib.sha256(guess).hexdigest()
    return guess_hash[:4] == "0000"
```

```
# Instantiate the Node
app = Flask(__name__)

# Generate a globally unique address for this node
node_identifier = str(uuid4()).replace('-', '')

# Instantiate the Blockchain
blockchain = Blockchain()



def mine():
    # Використовуємо функцію PoW для останнього блоку:
    last_block = blockchain.last_block
    proof = blockchain.proof_of_work(last_block)
    blockchain.new_transaction(
        sender="0",
        recipient=node_identifier,
        amount=1,
    )
    # Додаємо блок до блокчейну
    previous_hash = blockchain.hash(last_block)
    block = blockchain.new_block(proof, previous_hash)
    response = {
        'message': "Новий блок оброблено",
        'index': block['index'],
        'transactions': block['transactions'],
        'proof': block['proof'],
        'previous_hash': block['previous_hash'],
    }
    return jsonify(response), 200

@app.route('/transactions/new', methods=['POST'])
def new_transaction():
    values = request.get_json()
    # Перевірка на наявність всіх параметрів функції при запиті до серверу
    required = ['sender', 'recipient', 'amount']
    if not all(k in values for k in required):
        return 'Missing values', 400

    # Створення нової транзакції
    index = blockchain.new_transaction(values['sender'], values['recipient'], values['amount'])
    response = {'message': f'Транзакція буде добавлена до блоку {index}'}
    return jsonify(response), 201

@app.route('/chain', methods=['GET'])
```

```
def full_chain():
    response = {
        'chain': blockchain.chain,
        'length': len(blockchain.chain),
    }
    return jsonify(response), 200


@app.route('/nodes/register', methods=['POST'])
def register_nodes():
    values = request.get_json()

    nodes = values.get('nodes')
    if nodes is None:
        return "Error: Please supply a valid list of nodes", 400

    for node in nodes:
        blockchain.register_node(node)

    response = {
        'message': 'New nodes have been added',
        'total_nodes': list(blockchain.nodes),
    }
    return jsonify(response), 201


@app.route('/nodes/resolve', methods=['GET'])
def consensus():
    replaced = blockchain.resolve_conflicts()

    if replaced:
        response = {
            'message': 'Our chain was replaced',
            'new_chain': blockchain.chain
        }
    else:
        response = {
            'message': 'Our chain is authoritative',
            'chain': blockchain.chain
        }

    return jsonify(response), 200


if __name__ == '__main__':
    from argparse import ArgumentParser

    parser = ArgumentParser()
```

```
parser.add_argument('-p', '--port', default=5000, type=int, help='port to listen on')
args = parser.parse_args()
port = args.port

app.run(host='0.0.0.0', port=port)
```