Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Підвищення достовірності системи конфіденційного голосування на основі смарт-контрактів в мережі блокчейн

Виконав: студент 2-го курсу, групи УБ-21б
спеціальності 125– Кібербезпека
Освітня програма – Управління інформаційною безпекою
(шифр і назва напряму підготовки, спеціальності)

Манелюк В.Д.
(прізвище та ініціали)

Керівник: к.т.н., доц., доцент каф. МБІС
Карпінець В.В
(прізвище та ініціали)
«15» грудня 2022 р.

Рецензент: к.т.н., доц., доцент каф. ОТ
Крупельницький Л.В.
(прізвище та ініціали)
«15» грудня 2022 р.

Допущено до захисту
Голова секції УБ кафедри МБІС

Юрій ЯРЕМЧУК
“13” грудня 2022 р.

Вінниця ВНТУ - 2022 рік

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

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Управління інформаційною безпекою

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС

_____**Юрій ЯРЕМЧУК**
“ 15 ” вересня 2022 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу студенту
Манелюку Віталію Дмитровичу
(прізвище, ім`я, по-батькові)

1. Тема роботи Підвищення достовірності системи конфіденційного голосування на основі смарт-контрактів в мережі блокчейн

Керівник роботи к.т.н., доц., зав. Каф МБІС Карпінець В.В.
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “14” вересня 2022 року № 203
2. Строк подання студентом роботи 15 грудня 2022 р.
3. Вихідні дані до роботи: Стандарти, підручники та наукові статті по темі. Існуюче програмне забезпечення, яке стосується теми магістерської дипломної роботи.
4. Зміст текстової частини: Для досягнення поставленої мети необхідно розв'язати такі завдання: побудувати смарт-контракт на блокчейн мережі Ethereum; підключити до блокчейн мережі систему голосування з можливістю обробки даних у довіреному середовищі виконання (Trusted Execution Environment); налаштувати зв'язок між мережею та системою голосування, забезпечивши конфіденційність даних криптографічними методами; розробити графічний інтерфейс для взаємодії із системою голосування.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)
У першому розділі наведено 4 рисунки та 1 таблиці.
У другому розділі наведено 15 рисунків.
У третьому розділі наведено 11 рисунки.
У четвертому розділі наведено 6 таблиць.
У додатку Б Лістинг програмного коду смарт-контракту.
У додатку В наведено Лістинг програмного коду захищеного середовища.
У додатку Г наведено ілюстративний матеріал (презентація).
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина	Карпінець В. В., к.т.н., доц., зав. каф. МБІС		
Економічна частина	Лесько О. Й., , к.т.н., зав. кафедри ЕПВМ		

7.Дата видачі завдання 15 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1.	Початок роботи над МКР	15.09.2022	30.09.2022	
2.	Аналіз предметної області обраної теми	01.10.2022	10.10.2022	
3.	Апробація отриманих результатів	11.10.2022	15.10.2022	
4.	Розробка алгоритму роботи	16.10.2022	31.10.2022	
5.	Написання магістерської роботи на основі розробленої теми	01.11.2022	15.11.2022	
6.	Розробка економічної частини	15.11.2022	23.11.2022	
7.	Передзахист магістерської кваліфікаційної роботи	24.11.2022	25.11.2022	
8.	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	26.11.2022	18.12.2022	
9.	Захист магістерської кваліфікаційної роботи	19.12.2022	21.12.2022	

Студент _____ Манелюк В.Д.
(підпис)

Керівник роботи _____ Карпінець В.В.
(підпис)

АНОТАЦІЯ

Мета даної роботи полягає в підвищенні достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів. Об'єктом дослідження є технологія блокчейн, а також механізми передачі та обробки персональних даних, які забезпечують конфіденційність цих даних, а предметом – децентралізована система голосування та її рівень безпеки.

Завдання, які вирішувалися під час дослідження:

1. Розгляд технології децентралізованих мереж з прикладу Ethereum і механізму довіреного середовища виконання з прикладу Intel SGX.

2. Реалізація тестової мережі блокчейн із підключенням до неї зовнішнього середовища для надійної обробки персональних даних.

3. Опис смарт-контрактів для анклаву Intel SGX.

4. Тестування довіреного середовища.

Робота заснована на базі зібраних даних про методи побудови децентралізованих мереж та розробки додатків з використанням анклавів.

У ході роботи були виконані завдання: розгляд питань про розроблення децентралізованих додатків; проводився аналіз blockchain платформ; проектувався, розроблявся, удосконалювався та тестувався децентралізований додаток для інтернет-голосування.

ABSTRACT

The purpose of this work is to increase the reliability of the electronic voting system in the blockchain network with the protection of their confidentiality based on smart contracts. The object of research is blockchain technology, as well as the mechanisms of transfer and processing of personal data that ensure the confidentiality of this data, and the subject is the decentralized voting system and its level of security.

Tasks that were solved during the research:

1. Consideration of the technology of decentralized networks from the example of Ethereum and the mechanism of the trusted execution environment from the example of Intel SGX.

2. Implementation of a blockchain test network with connection to it of an external environment for reliable processing of personal data.

3. Description of smart contracts for the Intel SGX enclave.

4. Testing the trusted environment.

The work is based on the collected data on methods of building decentralized networks and developing applications using enclaves.

In the course of the work, the following tasks were completed: consideration of issues related to the development of decentralized applications; an analysis of blockchain platforms was carried out; designed, developed, improved and tested a decentralized application for Internet voting.

ЗМІСТ

ВСТУП

Актуальність. Інформаційні технології поступово дедалі переплітаються з людським життям. Перехід із реального світу до цифрового простору зазвичай тягне до себе дуже багато позитивних моментів.

Голосування - основа будь-якої успішної демократії і тому воно має бути доступним та безпечним для всіх людей. Сьогоднішні найпоширеніші паперові системи голосування доступні та дешеві, але мають дві основні проблеми. Згідно з багатьма експертами, такі системи не масштабуються (тому це призводить до таких основних проблемам, як точність), і вони мають на увазі «впевненість у процедурній безпеці організаторів, які проводять їх правильно і чесно» [1].

Нерідко в новинах описуються події про злам тієї чи іншої інформаційної системи, які дозволяють зловмисникам отримати несанкціонований доступ до конфіденційної інформації [2]. Актуальність роботи полягає в тому, що ці ризики можна мінімізувати завдяки стрімкому прогресу криптографії, в тому числі завдяки розвитку технології blockchain. Blockchain міг би запропонувати повсюдне масштабоване рішення поточних та застарілих виборчих методів, забезпечивши безпечне та захищене від фальсифікацій цифрове голосування.

Тому **метою цієї дипломної роботи є** підвищення достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів. Для досягнення поставленої мети необхідно розв'язати такі завдання:

- побудувати смарт-контракт на блокчейн мережі Ethereum;
- підключити до блокчейн мережі систему голосування з можливістю обробки даних у довіреному середовищі виконання (Trusted Execution Environment);
- налаштувати зв'язок між мережею та системою голосування, забезпечивши конфіденційність даних криптографічними методами;
- розробити графічний інтерфейс для взаємодії із системою голосування.

Об'єктом дослідження цієї випускної кваліфікаційної роботи є технологія блокчейн, а також механізми передачі та обробки персональних даних, які забезпечують конфіденційність цих даних.

Предметом дослідження є ступінь безпеки знайдених рішень, тобто наскільки запропоновані механізми запобігають крадіжці та розкриттю персональних даних.

Теоретичною основою випускної кваліфікаційної роботи стали дослідження компанії Intel у галузі довіреного середовища виконання Intel® Software Guard Extensions [3], проект Hyperledger Avalon та Hyperledger Fabric від компанії Linux Foundation [4].

Наукова новизна даної роботи полягає в наступному:

- створення механізму верифікації коду анклаву Intel SGX за допомогою смарт-контракту;

- надання можливості обробки даних без порушення їх конфіденційності.

У теоретичній частині розглядається питання щодо розробки децентралізованих додатків з урахуванням технології blockchain. Так само, проводиться порівняння між різними платформами blockchain. Підсумком цього порівняння є те, що найбільш повно сформульовано вимогам відповідає платформа Ethereum. Практична частина присвячена підвищення достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів.

В результаті виконаної роботи реалізовано децентралізовану систему електронного голосування із застосуванням смарт-контрактів на базі платформи Ethereum, в якій користувачі можуть висловлювати своє волевиявлення з того чи іншого питання з метою спільного пошуку, що задовольняє всіх учасників системи відповідного рішення із внесенням конфіденційних даних та впевненістю, що вони не будуть викрадені.

Захищеність конфіденційності персональних даних забезпечується криптографічними методами та покладається на прозорість мережі блокчейн. Цей механізм може застосовуватися для збору цінних даних від кількох

компаній, об'єднаних спільною мережею блокчейн та проводити аналіз цих даних за допомогою методів машинного навчання, отримуючи важливі статистичні та інші дані.

Практична частина роботи виконувалася на підставі документації до opensource проекту Ethereum, документації з мови програмування Solidity та документації з Intel® Software Guard Extensions [5].

При розробці випускної кваліфікаційної роботи використовувалися публікації та наукові видання, що зачіпають теоретичні та прикладні аспекти побудови блокчейн мереж, довіреного середовища виконання та закону про персональні дані.

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Розробка технології блокчейн є значним науково-технічним проривом [6]. З точки погляду інформаційної безпеки, принцип децентралізації забезпечує такі якості мережі як: доступність, незалежність та захищеність.

1.1 Паперове та електронне голосування

Голосування – це спосіб прийняття рішень групою людей, у якому думка вибирається шляхом підрахунку волевиявлень кожного із виборців [7].

Голосування може бути конфіденційним та відкритим. У відкритому голосуванні кожен учасник знає, хто і як розпорядився своїм голосом. При конфіденційному голосуванні, навпаки, вживаються заходи, щоб інформація про волевиявлення кожного з людей, що проголосували, була недоступна третім особам. У такому голосуванні підвищується ймовірність виключення можливості підкупу чи тиску на виборця[8]. Однак, у таємних голосуваннях неможливо визнати цю процедуру повністю чесною, так як такий спосіб виборів не можна верифікувати, якщо вони масштабні. Тому, нерідко на подібні процедури таємного паперового голосування можна зустріти різні маніпуляції, які можуть призвести до дискредитування результатів голосування.

Електронне голосування – процес ухвалення рішення, шляхом звичайного голосування із застосуванням спеціальних електронних засобів голосування та технічних електронних засобів для підрахунку голосів та оголошенням результату [9]. Один з різновидів такої процедури голосування є вибори через Інтернет. Технологія електронних голосуваннь дозволяє скоротити час, який потрібен для підрахунку голосів, а також полегшити процедуру голосування людям, яким не потрібно приходити на виборчу дільницю.

Голосування відіграє важливу роль у побудові демократичного товариства. Традиційне голосування вимагає, щоб виборці були присутні при процедурі голосування, що зазвичай тягне за собою витрати часу людей, які беруть участь

у виборах, а також дана процедура вимагає великих фінансових вкладень, якщо йдеться про вибори державного масштабу.

Електронне голосування - це нова концепція онлайн-виборів, заснована на криптографії. Система підтримує повнофункціональне онлайн-голосування на будь-яких пристроях, а результати опитування будуть розраховуватися автоматично та анонімно. Порівняно з традиційним голосуванням, електронне голосування – це економічніша система, більш прозора та неупереджена.

Система електронного голосування переважно використовує інтернет платформу. Найважливіша проблема для електронного голосування – це ризики безпеки, які можуть виникнути. З метою зниження ризиків, за останні десятки років було запропоновано безліч різних протоколів пов'язаних з конфіденційністю бюлетенів, індивідуальною перевіркою, надійністю, доступністю тощо. Крім того, у поданих протоколах реалізовані різні технології, такі як «сліпий підпис», «кільцевий підпис», «гомоморфне шифрування», «Mix networks», «доказ з нульовим розголошенням» тощо [10]. Та з появою криптовалюти, шифрування, відповідно, й інтернет-голосування стало досконалішим.

Таблиця 1 – Порівняння електронного та паперового голосування

	Паперове голосування	Електронне голосування
Висока швидкість обробки бюлетенів	-	+
Економія часу, при заповненні бюлетенів	-	+
Відображення ходу голосування у реальному часі	-	+
Ефективна масштабованість	-	+

Не дивлячись на позитивні сторони, у електронного голосування, по деяким міркам, ще більше ризиків фальсифікацій та компрометування результатів, ніж звичайного, так як система може бути зломана третіми особами,

а підрахунок голосів здійснюється на машині, доступ до якої є у адміністраторів, а значить, що вони також можуть мати доступ до результатів та змінювати їх на свій розсуд [11].

Проаналізувавши критику існуючих рішень систем електронного голосування, можна сформулювати такі недоліки таких систем та деякі сторони, які можна покращити:

- недостатня прозорість процесу голосування;
- недостатня стійкість до відмови системи;
- недостатня сійкість від злому та викраденню конфіденційних даних.

Виходячи із загальних вимог безпеки та відмовостійкості, все частіше пропонується протокол на основі технології Blockchain, який повинен позитивно позначитися на сформульованих вище недоліках.

Технологія, відома як «блокчейн» була вперше представлена Сатоші Накомото у його статті “Bitcoin: A Peer-to-Peer Electronic Cash System”, в якій викладено математичну основу для криптовалюти "Bitcoin" [12]. Хоча це була новаторська стаття, вона ніколи не представлялася в традиційному журналі, що рецензується, а справжня особистість автора невідома. Технологія блокчейн не лише лежить в основі всіх криптовалют, але і знаходить широке застосування у більш традиційній фінансової промисловості. Вона також відчинила двері для нових додатків, таких як "смарт-контракти".

Проблема, яку Накомото вирішив за допомогою ланцюжка блоків, полягала у встановленні довіри до децентралізованої (розподіленої) системи. Конкретніше проблема створення розподіленого сховища тимчасових файлів, у той час, як жодна зі сторін не може втручатися у зміст даних чи у відмітки часу без виявлення. Варто звернути увагу на те, що ця проблема ортогональна (паралельна) проблемам аутентифікації та цілісності, які вирішуються за допомогою цифрових підписів. Якщо сторона створює електронний підпис для документа, вона встановлює лише перевірюваний зв'язок між стороною та документом. Наявність дійсного цифрового підпису доводить, що сторона справді має намір підписати документ, і що документ не було змінено. Однак

цифровий підпис не гарантує нічого про час, в який документ було підписано: позначка часу вимагає довіри до сторони, яка її підписала. У разі фінансових операцій та інших форм юридичних договорів час має важливе значення, і порядок цих фінансових транзакцій має бути незалежно сертифікований для проведення аудиту [13].

Позитивні якості блокчейну та біткоїну можна розглянути на угодах, пов'язаних із нерухомістю. Власник може бути визначений як сторона, якою востаннє було продано будинок, але право власності може бути перевірено тільки з повного паперового дослідження всіх транзакцій, пов'язаних з будинком, до певної міри «паперовою доріжкою», яка зазвичай зберігається та перевіряється титульними компаніями. Варто відмітити, що система не повністю запобігає шахрайським транзакціям (наприклад, людина, яка продає будинок, яким вона не володіє, або продає те саме майно більш ніж одній стороні), але шахрайські дії зрештою виявляються, як і реальне право власності. Та ж перевірка власності виникає у фінансових транзакціях – обов'язково, при продажу криптовалюти, а також у продажу будь-якого іншого традиційного фінансового інструменту. Зазвичай проблема вирішується шляхом запису всіх транзакцій в одному довіреному централізованому регістрі, але «книга» не завжди є практичним рішенням, оскільки вона не масштабується для великої кількості частих транзакцій і тому вона вимагає, щоб усі сторони довіряли зберігачу книги. Так само вам потрібно довіряти своєму банку свої гроші (а співробітники банку, що крадуть кошти клієнтів не рідкість). Щоб усунути це, блокчейн надає механізм розподіленої довіри: кілька сторін зберігають записи транзакцій, і кожна сторона може перевірити, чи не було змінено порядок та позначки часу транзакцій [14].

1.2 Основні засади побудови децентралізованих мереж

Основний принцип децентралізованих мереж полягає у розподіленому реєстрі. Аналогічно бухгалтерській книзі, в яку записуються всі грошові перекази і нарахування, що здійснюються, в блокчейні зберігаються записи про проведені раніше операції. Ці записи можуть описувати як грошові переводи, так і укладені контракти. Копії реєстру з усією історією зберігаються у всіх учасників мережі. Блокчейн – це однорангова мережа, ресурси якої використовуються для підтвердження та схвалення кожної транзакції. У такій мережі немає центрального управління та відсутня можливість зміни та підробки реєстру. Таким чином, кожен учасник або вузол мережі має ту саму інформацію, що й усі інші учасники. При цьому будь-яка зміна стану мережі відбивається на кожному вузлі. Такий принцип унеможливлює підробку інформації, так як будь-які помилкові дані на одному з вузлів мережі не збігатимуться з даними інших вузлів, і така інформація сприйматиметься як недостовірна.

Записи в блокчейні організовані в «блоки», які пов'язані один з одним за допомогою криптографічної перевірки (рис. 1.1).

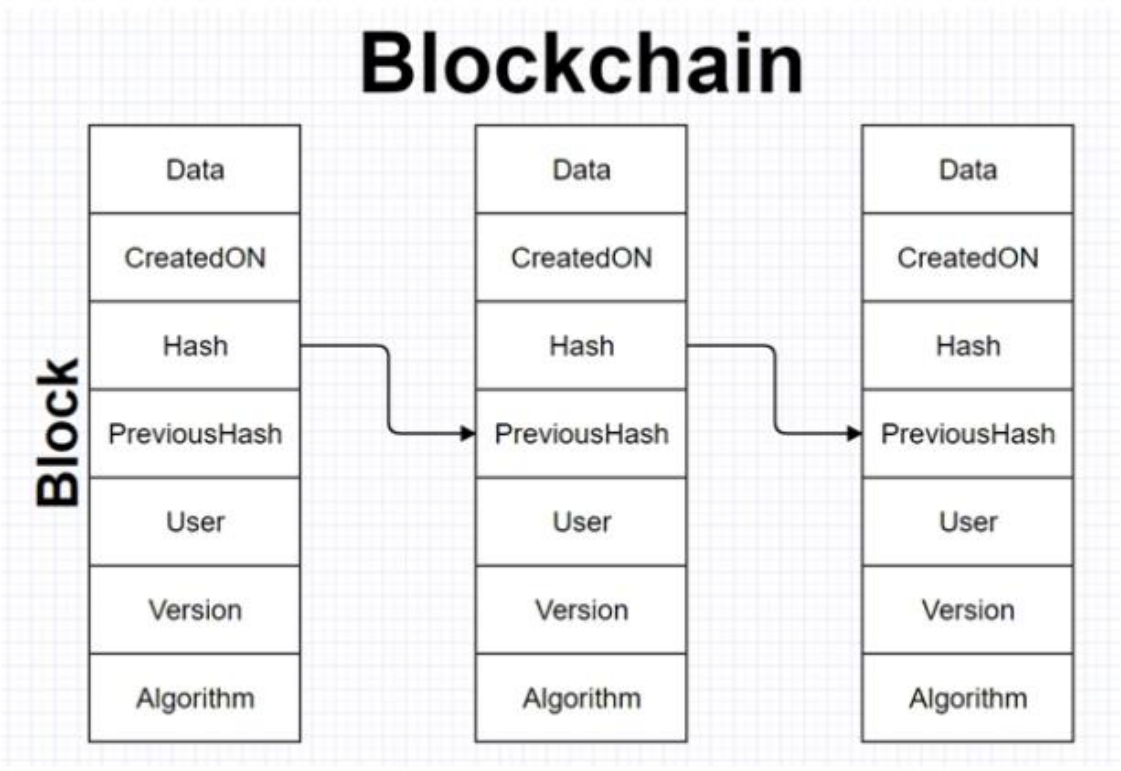


Рисунок 1.1 – Структура блоків в блокчейні

Кожна нова транзакція проходить перевірку учасниками, отримує схвалення та зберігається в блоці. Блоки формують майнери, вирішуючи складне математичне завдання. Зазвичай майнинг зводиться до серії обчислень з перебором параметрів для знаходження хешу із заданими властивостями, що вимагає великих операційних витрат [15]. Блок складається із заголовка, хеша заголовка попереднього блоку та певної кількості транзакцій. Блоки (blocks) поєднуються в ланцюг (chain), звідси і назва «blockchain», тобто ланцюг, що складається з блоків. Кожен блок дійсний лише тоді, коли він співвіднесений з попереднім. Даний принцип зв'язності виключає можливість вносити зміни до реєстру, оскільки для зміни одного блоку необхідно буде переписати всю його історію в блокчейні. Потрібно більше 50% від обчислювальної потужності доступної в блокчейні, щоб ефективно контролювати всю мережу, що в масштабах великої системи практично неможливо, так як для проведення «атаки 51%» у мережі біткоїн потрібно зламати 150 000 серверів одночасно [16].

Іншими словами - атака більшості, це те, що користувач або група контролює переважну більшість виробничих потужностей. Як результат, у них достатньо «сильних сторін» для контролю над великою кількістю подій. У цьому випадку створення нових блоків монополізується. Користувач або група, яка взяла монополію, винагороджується та не надається іншим для повної участі в процесі.

Розглянемо наступну ситуацію. Припустимо, Тетяна перераховує гроші через блокчейн-мережу, і вона стала жертвою 51% нападу, а Андрій - ні. У цьому випадку угода знаходиться в блоці, але зловмисники не дозволяють вам повернути гроші Андрія. Інші зловмисники формуватимуть блоки та створюватимуть транзакції, використовуючи потужність своїх комп’ютерів, але в цьому випадку мережа буде порушена, це зменшить потік клієнтів, що спричинить зменшення вартості валюти. В результаті нападник втратить лише свої гроші та свій час.

Загалом алгоритм перевірки завдань схожий на звітність на робочому столі. Співробітники регулярно подають аудиторські звіти, щоб підтвердити, що

вони виконали конкретне завдання. В іншому випадку вони не отримають зарплату, оскільки не підтвердили виконану роботу.

Є два види ланцюжків :

1) Public Blockchain - відкрита і сучасна база даних. Кожен учасник має право читати та записувати дані. Цей тип не підходить для організацій, що обробляють конфіденційну інформацію. Усі вузли рівні. Створюючи блоки, вони складають рядок і мають тимчасовий тег. Вони перевіряються комп'ютерами вузлів, перш ніж записуватися в блокчейн. Дані не є змінними, і всі транзакції є загальнодоступними. Цей тип Blockchain використовує криптовалюти Bitcoin та Ethereum. Однією з головних переваг публічного блокчейна є децентралізація. Наприклад, коли ми передаємо гроші з рук в руки, нам не потрібні банки чи посередники, це називається децентралізацією. Транзакція грошових переказів (в даному випадку в біткойнах) записується в мережу (блокчейн) раз і назавжди, змінити або скасувати її неможливо. Він не вимагає від посередників, які отримують гроші від клієнта і передають їх іншим, що гарантують укладення договору.

Рисунок 1.2 – Приватний Blockchain

Приватний Blockchain містить обмеження для читання/запису даних. Він використовується приватними організаціями і може співпрацювати лише за запрошенням цієї організації. Визначаються різні рівні доступу користувачів. Учасники повинні мати дозвіл читати, писати чи перевіряти сайт. Інформація

шифрується з метою конфіденційності. Приватний Blockchain набагато швидший, ніж загальнодоступний Blockchain, і дозволяє визначати пріоритетні вузли. Підвид Private Blockchain - це ексклюзивний блокчейн. Цей ланцюжок створює групу людей, що займаються обробкою транзакцій. Вона називається консорціум. Консорціум - приватна мережа яка включає декілька компаній. Контрольні вузли вибираються заздалегідь. Він також визначає правила перевірки блоків у головному ланцюзі та параметри доступу до мережі (Рисунок 1.3).

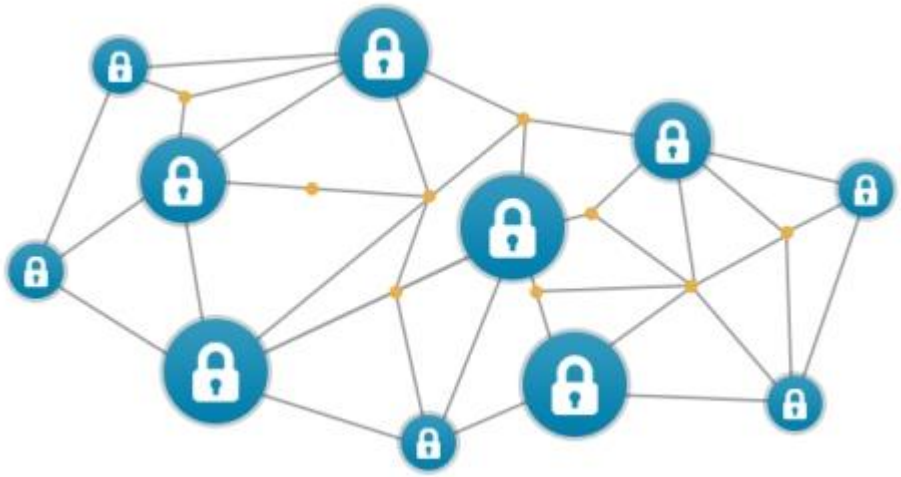


Рисунок 1.3 – Консорціум

Таким чином, можна виділити кілька основних принципів в основі побудови децентралізованих мереж:

1. Розподіл: блокчейн працює на комп'ютерах добровольців по всьому світу без центральної бази даних, яку можна було б зламати [17].

2. Публічність: будь-який користувач мережі може переглядати блокчейн у будь-який момент та бачити історію транзакцій у реєстрі.

3. Безпека: для великих мереж проведення «атаки 51%» практично неможливе через величезні обчислювальні витрати. Крім цього, в блокчейні використовується система шифрування, що застосовує публічні та приватні ключі для забезпечення конфіденційності даних.

При побудові децентралізованих мереж необхідно враховувати вище описані принципи насамперед. Сама по собі технологія блокчейн гарантує

безпеку та стабільність операцій, але при цьому також необхідно забезпечити правильну роботу використовуваних протоколів та налаштування компонентів мережі, що є завданням розробників

1.3 Принципи технології blockchain

Отже, технологія blockchain ґрунтується на 4 важливих принципах, які благотворно позначаться на застосуванні цієї технології у сфері електронного голосування.

1) Перевірка транзакцій у мережі з використанням криптографії.

На сьогоднішній день, у сферах, де використовується передача даних, після надсилання деякого повідомлення з відомостями, які потрібно синхронізувати з одержувачем, одержувач має оновити свій власний реєстр із даними. На сьогоднішній день не існує зручного, не витратного за часом та ефективного рішення, щоб звіряти такі копії. І саме технологія blockchain дозволяє впоратися з цією проблемою за допомогою різних способів. Наприклад, під час обміну одними і тими ж базовими даними або при наданні підтверджуючих елементів із метою верифікації даних. Користувачі blockchain реєстру досягають угоди щодо верифікації зміни даних використовують деякі алгоритми консенсусу, такі як Proof of Stake, Proof Of Work, Proof Of Activity, Proof of Weight тощо [18].

2) Розповсюдження копій інформації про транзакції серед великої кількості учасників системи.

База даних у системі на блокчейн або повністю, або частинами розподіляється між пристроями учасників системи. Це дозволяє уникнути критичних помилок, які могли б статися на одному пристрої. На даний момент, мультиплікування є досить великою проблемою для нинішніх технологій організації баз даних, та це призводить до позачергових складнощів та витрат у ході реалізацій різних проектів. Крім того, така перевага блокчейна дозволяє зберегти цілим усі копії, якщо в одній із них стався збій. Багато учасники можуть

також підтверджувати додавання нових транзакцій у ході перевірки зіставлення власними силами [19].

3) Децентралізований контроль доступу

У реєстрах з децентралізованих системах використовуються ключі та підписи з метою надати певним учасникам реєстру права на певні маніпуляції у загальній базі даних. Такі ключі можуть дозволити виконувати певні дії, які будуть можливі лише за дотримання деяких умов. Так, регулятор може мати ключ для читання, який дасть можливість перегляду інформації про транзакції відповідної організації. Але це можливо тільки за умови, якщо власник надасть регулятору відповідний ключ [20].

4) Конфіденційність та прозорість

Оскільки, велика кількість сторін бере участь у верифікації кожної транзакції, а копія даних знаходиться у багатьох учасників, така система забезпечує високий рівень прозорості. Ця властивість надає можливість переконатися, що реєстр не редагувався несанкціонованим шляхом. Кожна транзакція проводиться з використанням унікального цифрового підпису, завдяки чому підтверджується факт того, що певний користувач здійснив транзакцію відповідно до існуючих у системі правил [21].

Структура блоку в технології blockchain являє собою список транзакцій (тіло) та заголовка, який містить у собі ключі транзакцій.

Рисунок 1.4 – Зміст блоку

Тіло блоку містить у собі список усіх попередніх транзакцій.

Заголовок кожного блоку зберігає результат виконання хеш-функції поточної та попередньої транзакції. За допомогою хеша ідентифікатора блоку кожен блок зв'язується з попереднім та наступним блоком. Це технічне рішення дозволяє зберігати надійність системи.

Хеш ідентифікаторів розраховується з хеш транзакцій поточного блоку та хеша ідентифікатора попереднього блоку. Тобто в заголовку блоків зашифровано інформацію про дані з попередніх блоків. Звідси слід, що в такій системі є можливість відстежити всі коригування даних у кожному із блоків, так як якщо внести зміни до даних в одному блоці, зміниться ідентифікатор всіх наступних боків.

Виходить, маючи ланцюг із блоків, у яких міститься хеш будь-якого блоку, можна перевірити оригінальність ланцюга блоків, чи коректні дані в тому чи іншому блоці, чи присутні в ланцюзі підроблені блоки і т.п.

Щоб досягти угоди між учасниками системи щодо внесення змін до блокчейну використовуються алгоритми консенсусу, такі як proof-of-work (доказ виконання роботи) та proof-of stake (Доказ частки володіння).

Доказ виконання роботи (PoW) – це принцип захисту мережевих систем від зловживань послугами, наприклад, DoS-атак та інших видів атак. Даний метод заснований на вирішенні складної математичної завдання однією стороною, яка вимагає деяких обчислювальних потужностей та часу. А результат цього рішення може швидко перевірити інша сторона. Тобто особливістю таких обчислень є асиметричність витрат за часом. Ця особливість забезпечує сильний захист системи від фальсифікацій. Хеш ідентифікатора блоку повинен задовольняти певну умову, яка встановлює складність обчислень.

Доказ частки володіння (PoS) – принцип захисту, який заснований на доказі засобів, що шукаються. Цей метод є альтернативою proof-of-work. У разі використання цього принципу захисту, в ланцюжок блоків, з більшою ймовірністю, запишеться блок, обліковий запис якого має на балансі більше токенів.

Процес верифікації нових блоків називається майнінгом (mining). А учасник мережі блокчейн, який займається майнінгом називається майнером.

Майнер отримує від інших користувачів системи інформацію про нові записи у реєстр, збирає їх разом та намагається сформувати новий блок. Для цього йому потрібно вирахувати хеш ідентифікатора нового блоку. Наприклад, якщо після першого обчислення вийшов результат, який не задовольняє вимогам безпеки системи, у блоці існує спеціальне поле «nonce», яке спочатку дорівнює нулю. Так щоб отримати результат, який задовольнятиме вимогам безпеки, майнер робить перерахунок хеша зі зміненим полем "nonce" стільки разів, скільки знадобиться для того, щоб хеш вийшов відповідним.

Щоб обчислити хеш, який задовольнятиме вимогам, майнер найчастіше здійснює численні обчислення. І на той момент, коли відповідний хеш обчислений, блок верифікується та відправляється іншим учасникам мережі.

Процес майнінгу особливий тим, що незалежно від того, скільки повторних обчислень провів майнер, ймовірність обчислення потрібного рішення однакова у всіх. Ця особливість унеможливлює попередній розрахунок хешу або збереження результату всіх розрахунків, щоб використовувати їх надалі. Це робить усіх учасників майнінгу рівноправними.

За кожне успішне обчислення майнер отримує винагороду, комісію від транзакції, яку він розраховував. Ця винагорода та є причина, через яку він виконує таке трудомістке завдання [22].

1.4 Вимоги до голосування із застосуванням blockchain

В останні роки активно розвиваються децентралізовані цифрові валюти, у яких головними перевагами є захищеність від втручання третіх осіб у такі фінансові системи та прозорість. Деякі дослідники шукали спосіб голосування та підрахунку голосів за допомогою технології блокчейн. У 2015 році Дж. С. Чеплач в IT Університеті Копенгагена представив доповідь про сфери використання блокчейна де, у тому числі стверджував, що блокчейн може використовуватися для електронного голосування [23]. У той же час З. Жао та Т. Чан запропонували спосіб голосування з використанням технологій блокчейн та zk-SNARK. Такий спосіб голосування мав властивості конфіденційності, перевіряльності та незмінюваності [24]. У 2016 році було запропоновано протокол з використанням Zerocoin, який забезпечував би більшість вимог електронного голосування У тому ж році, П. С. Джейсон, та К. Ючі запропонували протокол з використанням сліпого підпису та карт Біткоїн [25].

Однак, використання Zerocoin складно реалізувати у програмному забезпечення у зв'язку з недостатнім літературним базисом та непопулярністю даної платформи. Zerocoin спеціалізується на розробці саме фінансових систем

та серед розробників децентралізованих додатків, не пов'язаних із фінансами, не має високого попиту.

Недоліками системи голосування, заснованої на платформі Bitcoin є низька швидкість транзакцій, через недостатню пропускну можливості її мережі [26]. Через надмірну популярність криптовалюти, черги транзакцій накопичується велика кількість транзакцій, як слідство, зростає час обробки транзакцій та вартість комісій [27].

Тому дані пропозиції не дозволяють реалізувати систему голосування на основі технології blockchain досить ефективною.

Отже, технологія blockchain, завдяки своїй децентралізованості, реплікації бази даних між учасниками, незмінності даних та збереження всіх тразакцій у вигляді ланцюжка блоків [28], що дозволяє використовувати ці позитивні якості в електронній системі голосування з метою усунення недоліків систем електронного голосування.

Система голосування на blockchain повинна виконувати наступні вимоги:

– можливість створити опитування та списки об'єктів голосування до них;

– реєстрація учасників для кожного опитування;

– можливість голосування;

– забезпечення прозорості;

– забезпечення відмовостійкості;

– відсутність можливості вносити несанкціоновані зміни, що впливають на результат голосування.

Тобто в такій системі у користувача має бути можливість створити потрібне опитування, після чого творцю опитування присуджуватиметься статус адміністратора опитування, за допомогою якого він зможе обмежувати коло осіб, які матимуть дозвіл на участь у створеному голосування.

Крім того, кожен користувач такої системи повинен мати можливість перегляду результатів голосування у реальному часі та перегляду ланцюжка блоків, щоб переконатися у прозорості голосування.

Крім цього, система повинна бути стійкою до відмови, тобто при виході з ладу пристрою з базою даних голосування система повинна продовжувати працювати, так як база даних буде реплікуватися на пристрої учасників мережі, в якій запущено децентралізовану програму.

Система також має бути злостійкою, тобто у зловмисника не повинно бути можливості впливати на перебіг голосування та його результати.

1.5 Висновки та постановка задач

Отже, технологія blockchain, завдяки своїй децентралізованості, реплікації бази даних між учасниками, незмінності даних та збереженню всіх тразакцій у вигляді ланцюжка блоків, дозволяє використовувати ці позитивні якості в електронній системі голосування з метою усунення недоліків систем електронного голосування.

Таким чином, у даному розділі випускної кваліфікаційної роботи було описано різницю між паперовим та електронним голосуванням, описані основні принципи побудови децентралізованих мереж та технології blockchain, а також описано основні вимоги до голосування із застосуванням blockchain. Тож мету даної роботи можна сформулювати так: підвищити достовірность системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів. Для досягнення поставленої мети необхідно розв'язати такі завдання:

- побудувати смарт-контракт на блокчейн мережі Ethereum;
- підключити до блокчейн мережі систему голосування з можливістю обробки даних у довіреному середовищі виконання (Trusted Execution Environment);
- налаштувати зв'язок між мережею та системою голосування, забезпечивши конфіденційність даних криптографічними методами;
- розробити графічний інтерфейс для взаємодії із системою голосування.

ПРОЕКТУВАННЯ СИСТЕМИ ЕЛЕКТРОННОГО ГОЛОСУВАННЯ З ПІДВИЩЕНОЮ ДОСТОВІРНІСТЮ

Основу конфіденційної системи електронного голосування складають два механізми: децентралізована мережа блокчейн і довірене середовище виконання. Поєднання цих двох технологій дозволить збільшити довіру до системи голосуваня і обробляти конфіденційні дані в безпеці.

2.1 Вибір платформи для системи захищенного голосування

На даний момент існує значна кількість реалізацій блокчейнів, серед яких є такі платформи, як Bitcoin, Ethereum, zCash, Waves, Ripple, Cordano, Stellar, IOTA та ін [29].

З перерахованих вище найпопулярніших платформ, такі платформи, як Ethereum, Waves і мають на меті створити платформи, що дозволяють вирішувати різні завдання за допомогою «смарт-контрактів», які забезпечують можливість різних маніпуляцій з даними в блокчейні.

Кожна з цих платформ має свої позитивні сторони, але на даний момент при розробці децентралізованих додатків найефективніше звернутися до Ethereum, оскільки інші платформи створені відносно недавно, мають меншу кількість навчальних матеріалів, а у засобах для розробки на даних платформах часто відбуваються зміни. Тому реалізація децентралізованих додатків на даних платформах буде скрутним та неефективним.

Наприкінці 2013 року канадський програміст Бутерін В. Д. представив громадськості єдину централізовану віртуальну машину, яка називалася Ethereum. Перший запуск мережі з відкритим кодом відбувся 30 липня 2015 року[30].

Автор платформи поставив собі за мету дати розробникам можливість реалізовувати різні децентралізовані програми, які мають властивості масштабованості, сумісності та простоти розробки. Досягнення мети дозволило

розробляти децентралізовані додатки та смарт-контракти, що мають формати транзакцій, правилами володіння та функціями зміни стану.

Велика популярність платформи Ethereum серед розробників та користувачів обумовлена, в тому числі тим, що такі алгоритми, як смарт контракти, вперше стали застосовуватися саме в проекті Ethereum.

Можливість виконувати різні маніпуляції з даними у ланцюжку блоків надається спеціальним комп'ютерним алгоритмом, який називається смарт контракт. Смарт контракт задає правила зберігання даних, описує набори функцій операцій над ними. Всі ці маніпуляції здійснюються за допомогою інтерфейсу, який надається смарт контрактом. Цей інтерфейс створюється з вихідного коду окремо від компіляції, а потім надає можливість виконувати двійковий код. Як і передбачається блокчейну, всі дані в Ethereum надаються кожному за учасника, так як. вони реплікуються та розподіляються між користувачами мережі. Внесення змін відбувається у вигляді транзакцій. У Ethereum конструкція транзакції виглядає наступним чином:

- Одержувач.
- Електронний підпис відправника.
- Сума переказу у валюті «ETH», яка використовується в системі як
- токен.
- Коментар.
- Ліміт "палива" на транзакцію (gasLimit).
- Вартість одиниці "палива" (gasPrice).

«Паливо» - це одиниця виміру, яка використовується для визначення оплати за конкретним обчисленням. Вартість «палива» - це кількість ETH, яку ви можете витратити на одиницю палива. Вартість "палива" вимірюється в "gwei". 1 Gwei дорівнює 0,000000001 ETH.

Для всіх транзакцій відправник повинен встановити ліміт палива та його вартість. Ціна на паливо та ліміт палива – це максимальна сума у Gwei, яку відправник готовий заплатити за переклад токенів.

Так, наприклад, встановивши gasLimit в 47 000 gwei, а gasPrice в 25 gwei, відправник готова заплатити 1175000 gwei, або 0,001175 ETH.

Тобто для здійснення транзакції потрібні витрати «палива», які виплатяться майнеру. Підтвердження транзакції відбувається на пристрої майнера, у віртуальній машині Ethereum Virtual Machine, де виконується смарт контракт.

Децентралізована програма (Dapp) – програма, яка запускається учасниками децентралізованої мережі із протоколами захисту (PoW, PoS)[31]. Програма може базуватися на різних технологіях, але зараз під Dapp, в основному, мається на увазі додаток на blockchain з використання смарт контрактів. Логіка сучасного Dapp це робота смарт контракту та UI для взаємодії з додатком. Відправлення більш-менш об'ємних даних та обмін повідомленнями в ідеальному DApp теж мають бути децентралізованими, проте ці технології лише починають з'являтися і заслуговують на окрему статтю. Блокчейн же забезпечує зберігання поточного стану та реалізує бізнес-логіку через смарт-контракти.

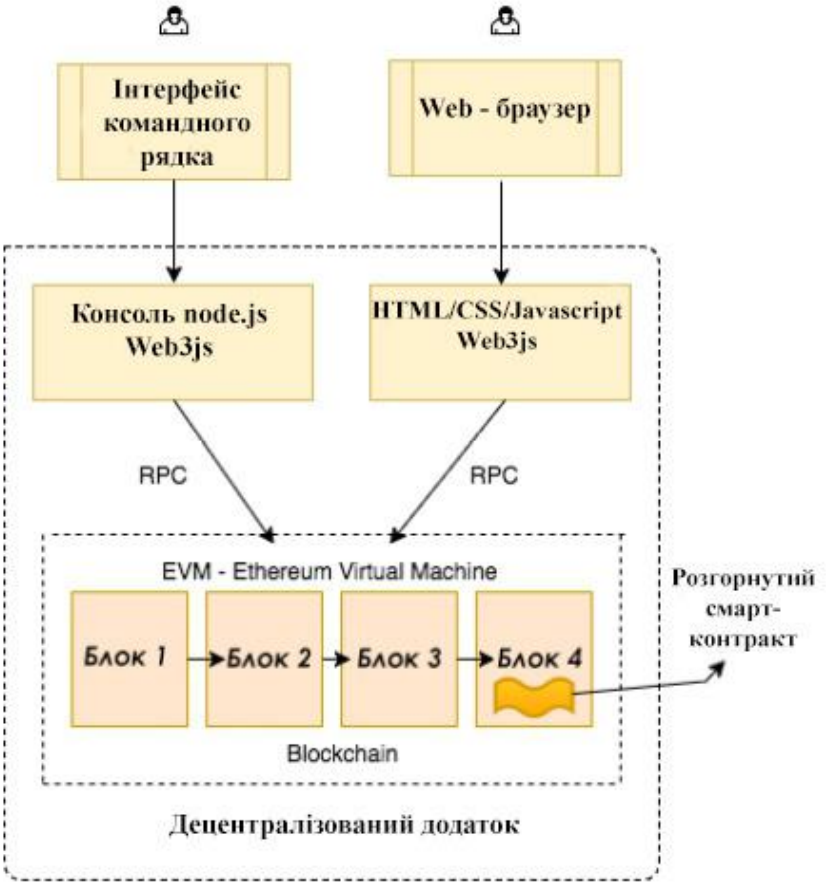


Рисунок 2.1 – Взаємодія з Ethereum Dapp

На рисунку 2.1 представлений огляд взаємодії користувача з децентралізованим додатком.

Взаємодіючи з Ethereum Dapp, користувач має можливість звертається безпосередньо до блокчейну через спеціальне програмне забезпечення або через інтерфейс командного рядка на своєму пристрої.

Для розробки смарт контракту для платформи Ethereum найчастіше використовується об'єктно-орієнтована, предметно-орієнтована мова програмування Solidity. Ця мова програмування була представлена у 2014 році та рекомендується до використання командою проекту Ethereum, в офіційному репозиторії платформи Ethereum міститься документація та посібник з даної мови програмування.

Solidity – високорівнева мова для EVM із синтаксисом, наближеним до мови програмування JavaScript. Його схожість з JavaScript дозволяє швидко

адаптуватися до написання смарт-контрактів розробникам, які займаються веб-програмуванням[32].

Оскільки ця мова рекомендується розробниками проекту Ethereum, мною було вирішено вибрати саме цю мову для написання смарт-контракту.

В офіційному репозиторії Ethereum, у посібнику до мови програмування Solidity також додається посилання на інтегровану середовище розробки під назвою Remix Solidity.

Remix – найпопулярніше середовище розробки смарт-контрактів. Її популярність обумовлюється його простотою. Так, щоб почати написання договору мовою Solidity, з цим середовищем розробки не потрібно ніяких складних маніпуляцій. Вона працює в браузері, і для того, щоб почати розробку, достатньо зайти на веб-сайт remix.ethereum.org, де на розробника вже чекає відкритий текстовий редактор, у якому можна починати писати програмний код.

Remix підтримує безліч функцій, корисних для розробки Dapp, наприклад, такі як:

- Покроковий відладчик;
- Підрахунок вартості "палива" для виконання функцій;
- Компіляція смарт-контракту;
- Підключення до бажаного сервера RPC;
- Та інше.

Крім середовища програмування для розробки Dapp також може знадобиться програмна платформа node.js, яка має безліч корисних бібліотек для розробки Dapp[33].

Під час розробки системи електронного голосування ефективно буде застосовувати інструмент Ganache-CLI, який дозволяє з однієї команди підняти у себе на пристрої симулятор блокчейна, з увімкненим RPC протоколом. Цей інструмент створює 10 пробних облікових записів у даному блокчейні. Баланс кожного тестового облікового запису має 100 ETH. Дана процедура дозволяє прискорити тестування написаної програми, так як відсутня потреба витрачати

час на підняття реального приватного блокчейна, створення облікових записів і т.д.

Крім Ganache-CLI, node.js має таку важливу для розробки бібліотеки, як Web3.js. Ця бібліотека дозволяє використовувати API ефіру за допомогою звичайного JavaScript.

2.2 Проектування системи для децентралізованого голосування

Розробка програми починається з його проектування. Проектування додатку, що розробляється, має на увазі під собою продумування базових функцій, яке матиме електронне голосування та повноваження користувачів.

Проектований додаток для електронного голосування має відповідати деяким функціональним вимогам. На рисуноку 2.2 представлена модель системи голосування, де показано основні функції програми.

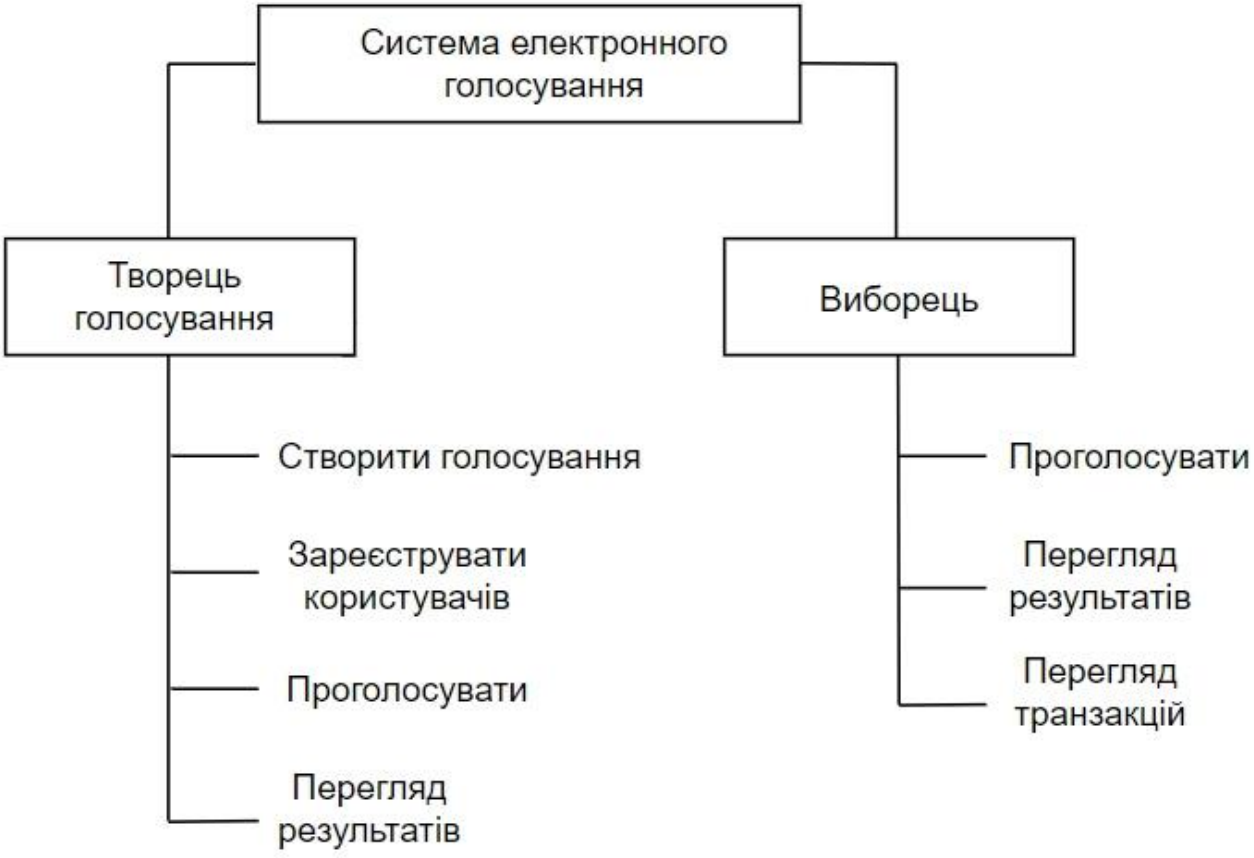


Рисунок 2.2 – Модель розробленого додатку

Основні функції, що надає система голосування:

– створити опитування та бути його адміністратором з можливістю реєстрації виборців;

– віддавати свій голос із внесенням власних даних за той чи інший об'єкт голосування;

– переглядати результати голосування;

– переглядати пов'язані зі смарт-контрактом транзакції, щоб переконатися у їхній чесності.

При зверненні до смарт контракту він дозволяє користувачеві створити опитування, передавши масив із елементами голосування, такими як: назва опитування, питання, варіанти голосування. Після створення опитування він може реєструвати нових користувачів у своєму опитуванні, які після реєстрації матимуть змогу проголосувати у цьому голосуванні. Також, смарт контракт надає можливість звернутись до блоку із даними про результати голосування. Крім того, користувач має можливість переглянути історію транзакції, таку можливість надає мережа блокчейн, у якій розгорнутий смарт-контракт.

Логічне моделювання є здійсненням перевірки функціонування логічної схеми.

Основна мета полягає у здійсненні перевірки функцій проектованого додатка без повної реалізації на даному етапі розробки. Переваги даної моделі полягають в тому, що здійснюється перевірка, як логічних функцій додатку, так і його тимчасові співвідношення. Для здійснення моделювання необхідно побудувати діаграму послідовності та діаграму варіантів використання програми.

На малюнку 2.3 показано діаграму послідовності, на якій відображено взаємодію об'єктів.

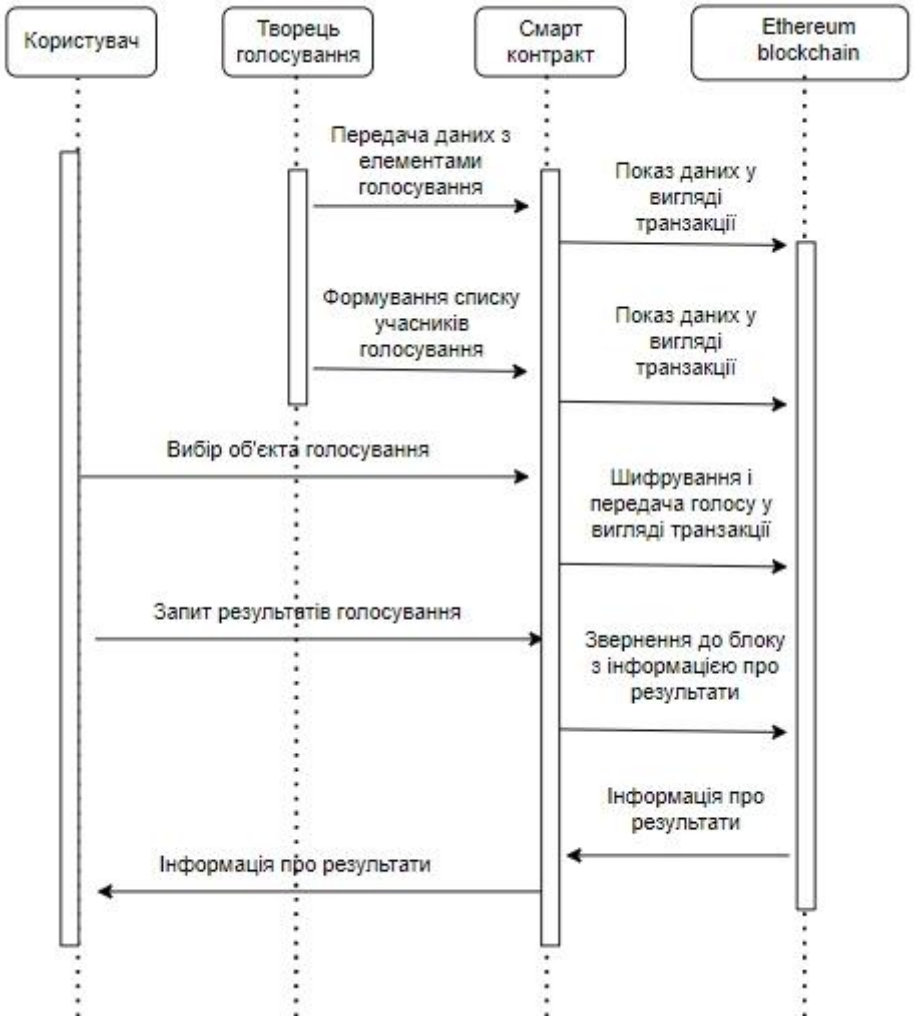


Рисунок 2.3 – Діаграма послідовності роботи системи електронного голосування

Взаємодія між об'єктами та суб'єктами відбувається наступним чином:

1. До розробленого смарт-контракту звертається творець нового голосування та створює нове опитування.

2. Такі дані, як «Назва голосування», «Обговорюване питання», «Варіанти голосування» формуються в блок і відправляються до мережі blockchain.

3. Автор голосування реєструє виборців у систему, шляхом додавання адрес їх облікових записів до списку виборців, що дає їм право на голосування у цьому опитуванні.

4. Сформований список виборців надсилається до blockchain.

5. Користувач звертається до смарт-контракту та обирає в опитуванні голосування.

6. Ім'я об'єкта голосування, за якого було віддано голос шифрується та відправляється в блокчейн.

7. Користувач звертається до смарт контракту, щоб дізнатися про результати голосування.

8. Смарт-контракт звертається до потрібного блоку з інформацією в блокчейні.

9. Інформація про результати голосування відправляються в захищене середовище обробки.

Таким чином, було здійснено перевірку логіки функціонування програми. Описано взаємозв'язок та послідовність роботи у додатку.

На малюнку 2.4 представлено діаграму варіантів використання системи голосування користувачем.

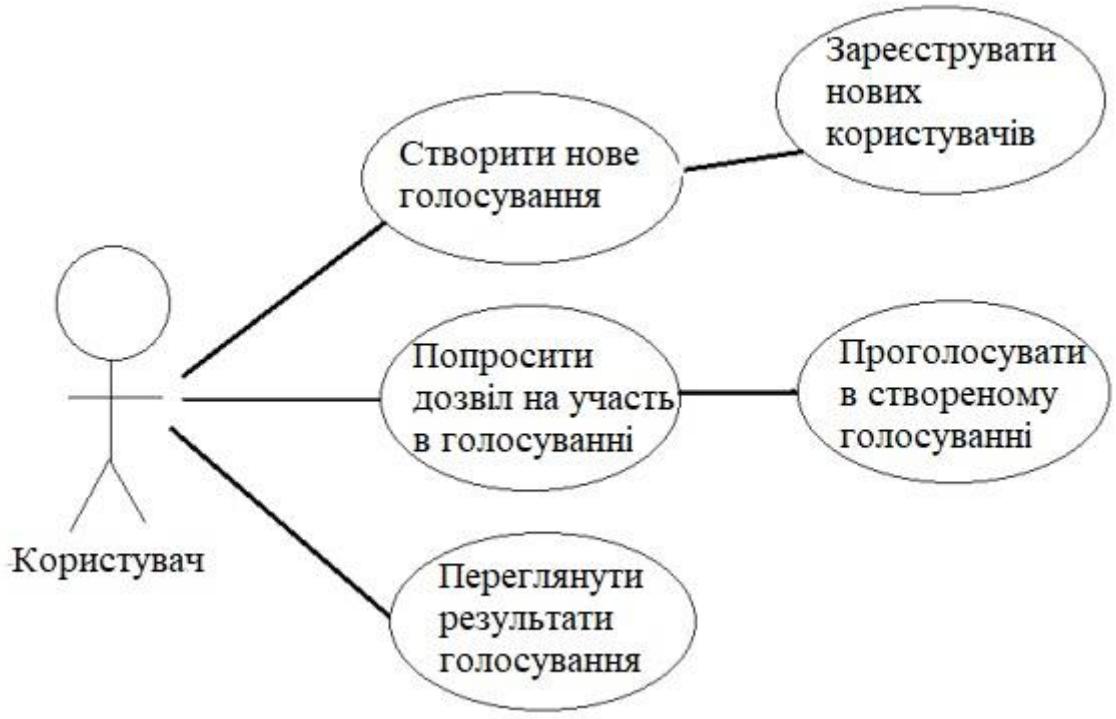


Рисунок 2.4 – Діаграма варіантів використання

Після того, як отримано уявлення про основні функції системи голосування, можна розпочати її реалізацію.

2.3 Розробка смарт-контракту для голосування

Після того, як вибрано всі необхідні засоби для реалізації, можна розпочати його написання.

Для початку необхідно сформувати базові вимоги до смарт контракту, тобто визначитися, які маніпуляції з даними в блокчейн він дозволяє виконувати і далі, відштовхуючись від цих функцій, писати програмний код смарт-контракту.

На самому початку у смарт-контракті потрібно буде реалізувати 3 простих функції:

– конструктор, якому ми передаватимемо масив об'єктів, за які йтимуть голосування;

– надання автору опитування статусу адміністратора;

– функція, яка дозволить голосувати за потрібний об'єкт;

– шифрування вибраного імені об'єкта голосування для конфіденційності вибору;

– функція, яка повертає кількість голосів того чи іншого голосування.

– Функція, яка відправляє дані на захищене середовище.

У випадку системи голосування, що розробляється, на початку контракту використовується структура, тобто комплексний тип даних, до яких будуть посилатися на основні функції.

```
struct Start {
    string title;
    string question;
    string option1;
    string option2;
    uint count1;
    uint count2;
    mapping (address => bool) voted;
    bool exists;
}
```

Рисунок 2.5 – Структура, до якої будуть посилатися функції

У цій структурі містяться такі дані, як «Назва опитування», «Обговорюване питання», «Об'єкт голосування 1», «Об'єкт голосування 2», «Об'єкт голосування 3», лічильники голосів за кожного з кандидатів, статус про те, проголосував той чи інший акаунт, чи ні чи створено опитування чи ні.

Далі йде конструктор, який викликається під час розгортання смарт-контракту, або в момент, коли необхідно створити нове опитування, який передаватиметься масив з об'єктами голосування, назвою опитування та питання, що обговорюється, тобто всі елементи голосування, описані у структурі.

```
function createVoting (string _question, string _name, string _option1, string _option2, string _option3) public {
        polls[_name].title = _name;
        polls[_name].question = _question;
        polls[_name].option1 = _option1;
        polls[_name].option2 = _option2;
        polls[_name].option3 = _option3;
        polls[_name].count1 = 0;
        polls[_name].count2 = 0;
        polls[_name].count3 = 0;
        polls[_name].exists = true;
    }
```

Рисунок 2.6 – Конструктор, до якого передається інформація про елементи голосування

Далі, створено функцію, яка дозволяє додати голос виборця до лічильника голосів об'єкта голосування із застосуванням протоколу шифрування SHA-3.

```
function vote (string optionName, string pollName) public {
    require(doesPollExist(pollName));
    require(hasAlreadyVoted(pollName));

    polls[pollName].voted[msg.sender] = true;

    if (keccak256(polls[pollName].option1) == keccak256(optionName)) {
        polls[pollName].count1 += 1; }
     if (keccak256(polls[pollName].option2) == keccak256(optionName)) {
        polls[pollName].count2 += 1; }
    if (keccak256(polls[pollName].option3) == keccak256(optionName)) {
        polls[pollName].count3 += 1; }
}
```

Рисунок 2.7 – Функція «Голосувати за»

Ця функція дозволяє додати голос до того чи іншого об'єкта голосування, попередньо перевіривши 2 умови:

1) чи існує опитування, у якому бере участь виборець;

2) чи була задіяна ця функція раніше з цього облікового запису.

Після перевірки цих умов функція присуджує голосуючому акаунту статус «проголосував» і додає один голос до лічильника одного із трьох об'єктів голосування.

Описані у функції голосування умови, що визначаються функцією, яка перевіряє чи запитуване опитування функцією, яка перевіряє статус виборця, чи було здійснено передачу голосу об'єкту голосування, чи ні.

```
function doesPollExist (string pollName) private view returns (bool) {
    if (polls[pollName].exists) {
        return true;
    } else {
        return false;
    }
}
```

Рисунок 2.8 – Функція перевірки наявності опитування

```
function hasAlreadyVoted (string pollName) private view returns (bool) {
    if (polls[pollName].voted[msg.sender]) {
        return false;
    } else {
        return true;
    }
}
```

Рисунок 2.9 – Функція перевірки статусу виборця

Функцію перегляду лічильника голосів об'єкта голосування реалізовано на рисунку 2.10.

```
function getPollCounts (string pollName) public view returns (uint[3]) {
    require(doesPollExist(pollName));

    return [polls[pollName].count1, polls[pollName].count2, polls[pollName].count3];
}
```

Рисунок 2.10 – Функція, що виводить кількість голосів за об'єкти голосування

Також у смарт-контракті реалізовані функції, що дозволяють переглянути варіанти голосування в тому чи іншому опитуванні.

```
function getOption1 (string pollName) public view returns (string) {
    require(doesPollExist(pollName));

    return polls[pollName].option1;
}

function getOption2 (string pollName) public view returns (string) {
    require(doesPollExist(pollName));

    return polls[pollName].option2;
}

function getOption3 (string pollName) public view returns (string) {
    require(doesPollExist(pollName));

    return polls[pollName].option3;
}
```

Рисунок 2.11 – Функція для виведення назви об'єктів голосування в створених опитуваннях

Після того, як отримано уявлення про основні функції системи голосування, можна розпочати реалізацію захищеного середовища.

2.4 Реалізація захищеного середовища для голосування

Компанія Intel у 2015 році представила свою технологію з використанням TEE – Intel Software Guard Extensions (Intel SGX)[34].

Intel SGX – це набір інструкцій центрального процесора, що надає можливість створювати анклави. Анклави – області у віртуальному адресному просторі, захищені від читання та запису іншими процесорами (включаючи ядро ОС) які не з цієї області [35].

Intel SGX створювалася для безпечних віддалених обчислень, тобто для запуску програмного забезпечення на віддаленому комп'ютері потенційно ненадійної сторони, при цьому з гарантіями цілісності та конфіденційності інформації[36].

Intel SGX забезпечує наступні засоби захисту від апаратних та програмних атак [37]:

- читання з анклаву та запис в анклав заборонені ззовні, незалежно від рівня привілеїв та режиму центрального процесора;

- налагодження експлуатованих анклавів недоступне, але розробник може встановити можливість налагодження під час процесу розробки;

- Спілкування з анклавом відбувається за допомогою спеціальних функцій ECALL (Enclave Call) та OCALL (Outside Call). Ніякі інші системні виклики та операції з регістрами не пов'язують основну ОС з анклавом;

- пам'ять анклаву зашифрована з використанням алгоритмів шифрування з відкритим ключем;

- ключі шифрування змінюються випадково при включенні живлення або перезавантаження системи. Ключі зберігаються в ЦП в ізольованому сховищі (KeyStorage);

- доступ до даних, ізольованих усередині анклавів, може бути отриманий тільки за допомогою коду, який використовує цей анклав.

Основний чейнкод мережі включає смарт-контракт PDnSC. Він відповідає за прийняття персональних даних від peers, збереження їх у форматі JSON у базі даних у вигляді ключ-значення, встановлення загального секретного ключа з анклавом SGX, перевірку хеша анклаву, шифрування персональних даних на відкритому ключі, відправку даних на захищенне середовище, розшифровку та запис результатів обробки до бази даних, з якої peers можуть отримати результати у відкритому вигляді.

Код смарт-контракту PDnSC представлений у додатку 1. Для роботи було наведено приклад оцінки ефективності використання вакцини від Covid-19.

Смарт-контракт має таку структуру:

Визначення структури з 5 полями для зберігання бази даних за ключом, вказаному у структурі Records:

```
type Records struct {
    Fullname string `json: "fullname"` // повне ім'я пацієнта.
    DOB string `json:"dob"`// Дата народження.
    Insurance string `json:"insurance"` // номер страховки з
```

вказівкою страхової фірми.

Vaccine string `json:"vaccine"` // назва використовуваної вакцини.

Status string `json:"status"`// стан після застосування вакцини.

}

Функція ініціалізації initLedger використовується при установці чейнкоду, приймає на вхід контекст виконання чейнкоду та ініціалізує базу даних початковими значеннями:

func (s * SmartContract) initLedger(APIstub shim.ChaincodeStubInter face) sc.Response

Функція recordPD додає в базу даних нові записи від peers, приймає на вхід контекст виконання чейнкод і самі дані у вигляді масиву рядків:

func (s * SmartContract) recordPD (APIstub shim.ChaincodeStubIn terface, args []string) sc.Response

Функція queryPD виводить записи з бази даних за ключем, приймає на вхід контекст виконання чейнкод і ключ, по якому знаходяться дані:

func (s * SmartContract) queryPD(APIstub shim.ChaincodeStubIn terface, args []string) sc.Response

Функція changPD дозволяє редагувати дані користувачів, замінюючи поля в базі даних, на вхід приймає контекст виконання чейнкоду, ключ за яким знаходяться дані та нові дані для заміни:

func (s * SmartContract) changePD(APIstub shim.ChaincodeStubIn terface, args []string) sc.Response

Функція CheckHash порівнює хеш зі значенням, що зберігається в базі даних, на вхід приймає контекст виконання чейнкоду, ідентифікатор захищенного середовища і порівнюваний хеш:

func (s * SmartContract) checkHash(APIstub shim.ChaincodeStubInter face, args []string) sc.Response

Для шифрування підключено додатковий модуль, з якого використовують методи Decrypter для розшифрування та Encrypter для шифрування.

Ми можемо довіряти виробнику апаратного забезпечення на віддаленому комп'ютері та передавати свої дані в захищений контейнер, розміщений на довіреному устаткуванні (рис. 2.12) [38].

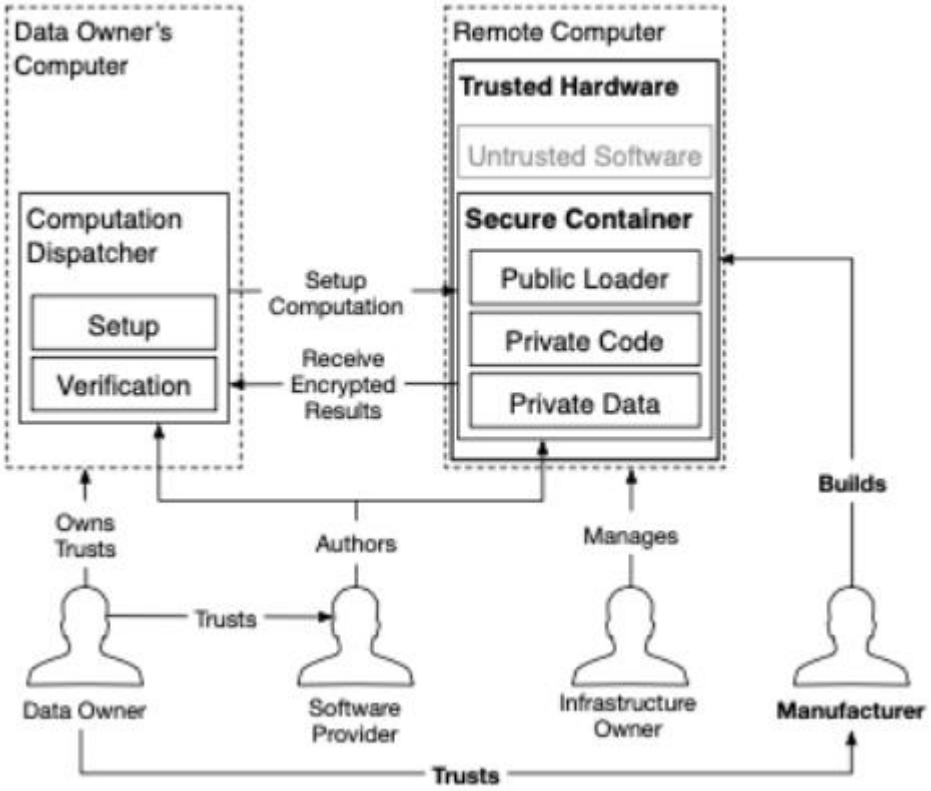


Рисунок 2.12 – Схема обробки даних на віддаленому комп’ютері

Дана схема може прийнятись в різних областях, не лише при голосуванні. Так, наприклад, у медичній сфері може бути реалізований хмарний сервіс, який виконує обробку конфіденційних медичних зображень для пошуку потенційних хвороб у пацієнтів [39]. Зображення у зашифрованому вигляді передаються на віддалений комп'ютер із Intel SGX. Анклав містить код для розшифровки зображень, алгоритм обробки зображень та код для шифрування результатів. За межами анклаву виконується код, який отримує завантажені зашифровані зображення і зберігає їх (Untrusted Part).

Процесор з підтримкою Intel SGX захищає цілісність і конфіденційність обчислень всередині анклаву, ізолюючи код і дані анклава від зовнішнього середовища, включаючи операційну систему та гіпервізор, а також апаратні пристрої, підключені до системної шини. У той же час модель SGX залишається

сумісною з традиційними рівнями програмного забезпечення в архітектурі Intel, де ядро ОС та гіпервізор управляють ресурсами комп'ютера.

Більшість комп'ютерів працюють на процесорах Intel та AMD. В Intel реалізовано різні рівні привілеїв – кільця привілеїв (рисунок 2.13).

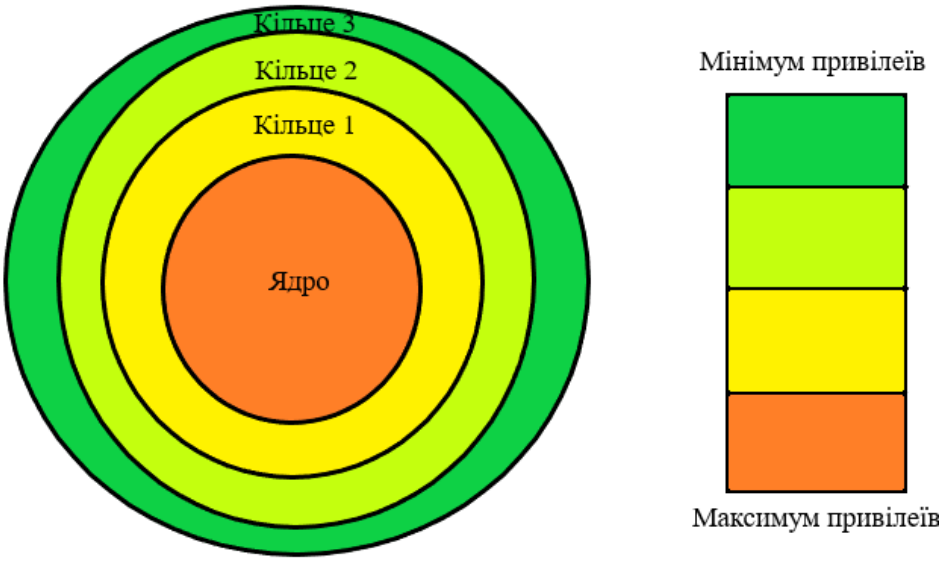


Рисунок 2.13 – Кільця привілеїв

Привілеї визначають дозволені комп'ютерні операції та доступ до пам'яті в будь-який момент часу і використовуються для забезпечення безпеки у комп'ютерній системі. Значення 0 відповідає найвищому рівню привілеїв, а 3 найнижчому [40]

- кільце 0: зазвичай ядро операційної системи та призначене для виконання найкритичніших програм;

- Кільце 1: драйвера пристроїв;

- Кільце 2: операційні системи;

- кільце 3: користувацькі програми.

Використання всіх чотирьох рівнів привілеїв не є обов’язковими. Також можливе додавання нових кілець привілеїв для обмеження повноважень компонентів ієрархії, але ускладнення архітектури зазвичай не робить систему більш захищеною, а навпаки, чим складніша система, тим більше ймовірність наявності у ній вразливості. Запропонований механізм роботи з анклавами допомагає створювати безпечні програми з мінімальними витратами, не змінюючи архітектури системи.

Додатки, що використовують Intel SGX, складаються з двох частин: довірена (trusted) та ненадійна (untrusted) частина.

Довірений компонент – це анклав. Код у надійній частині звертається до секретів програми. Програма може мати більше одного довіреного анклава.

Ненадійний компонент – це решта програми та будь-які його модулі. З погляду анклаву, ОС та VMM вважаються ненадійними компонентами.

Довірений компонент повинен обмежуватися даними, які потребують максимального захисту, і тими операціями, які мають діяти безпосередньо на них. Анклав великих розмірів вимагатиме більше пам'яті та збільшить вірогідність атаки.

Анклави повинні мати мінімальну взаємодію між довіреними та ненадійними компонентами [41]. Хоча анклави можуть залишати захищену область пам'яті та викликати функції в ненадійному компоненті (за допомогою спеціальної інструкції), обмеження цих залежностей застережить анклав від атак.

Процес виконання програми Intel SGX представлений малюнку 2.6

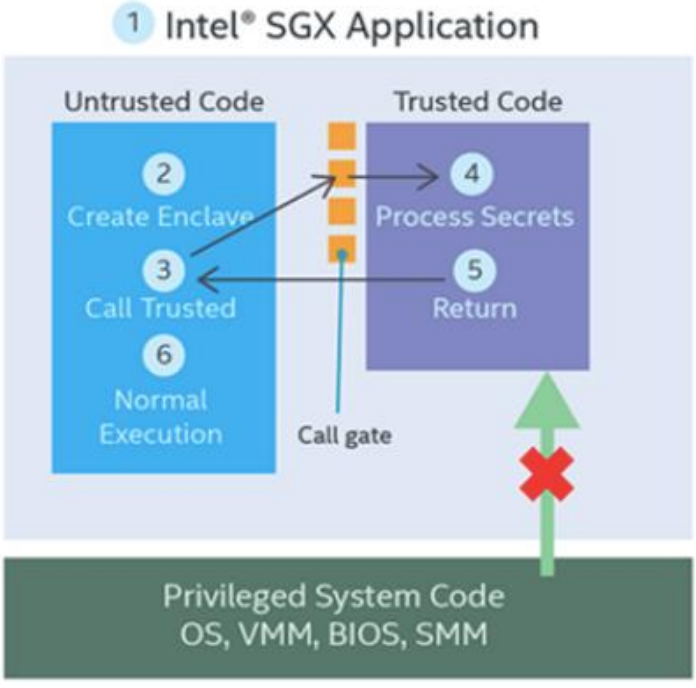


Рисунок 2.14 – Процес виконання програми Intel SGX

Під час виконання програми з Intel SGX можна виділити такі етапи:

1. Поділ програми на два компоненти: довірений та ненадійний.

2. Створення анклаву у захищеному компоненті.

3. Передача виконання коду анклаву за допомогою довіреного виклику функції з ненадійної частини програми.

4. Код анклаву обробляє дані у відкритому вигляді, при цьому доступ до них із привілейованого рівня ОЗ заборонено.

5. Передача результатів роботи коду анклаву в ненадійну частину програми.

6. Виконання у звичайному режимі у ненадійному компоненті.

Використовуючи довірені обчислювальні ресурси поза мережею, ми можемо збільшити пропускну спроможність та покращити конфіденційність даних. Збереження цілісності виконання та забезпечення дотримання гарантій конфіденційності досягається за рахунок використання опції довірених обчислень у Trusted Execution Environment.

Служба довірених обчислень (TCS) розміщує довірених робітників (Trusted Workers) та робить їх доступними для виконання робочих завдань (Work Orders), відправлених запитуючими через інтерфейс користувача або інструменти командного рядка. Замовлення виконання робіт також можуть бути надіслані за допомогою смарт-контрактів (залежно від корпоративного додатка), що працюють на DLT (рисунок 3.1).

Компоненти архітектури Hyperledger Avalon [42]:

- worker registry: тут перераховані довірені обчислювальні працівники. Він включає звіт про перевірку атестації, відкритий ключ шифрування RSA і відкритий ключ перевірки ECDSA SECP256K1;

- workload: включає сервіс атестації (IAS), Key Management Enclave, який має доступ до закритих ключів підпису та шифрування працівника;

- front-end application: зовнішня програма може бути інтерфейсом користувача або сценарієм, який використовується для ініціювання запитів на робочі завдання та отримання відповідей на них.

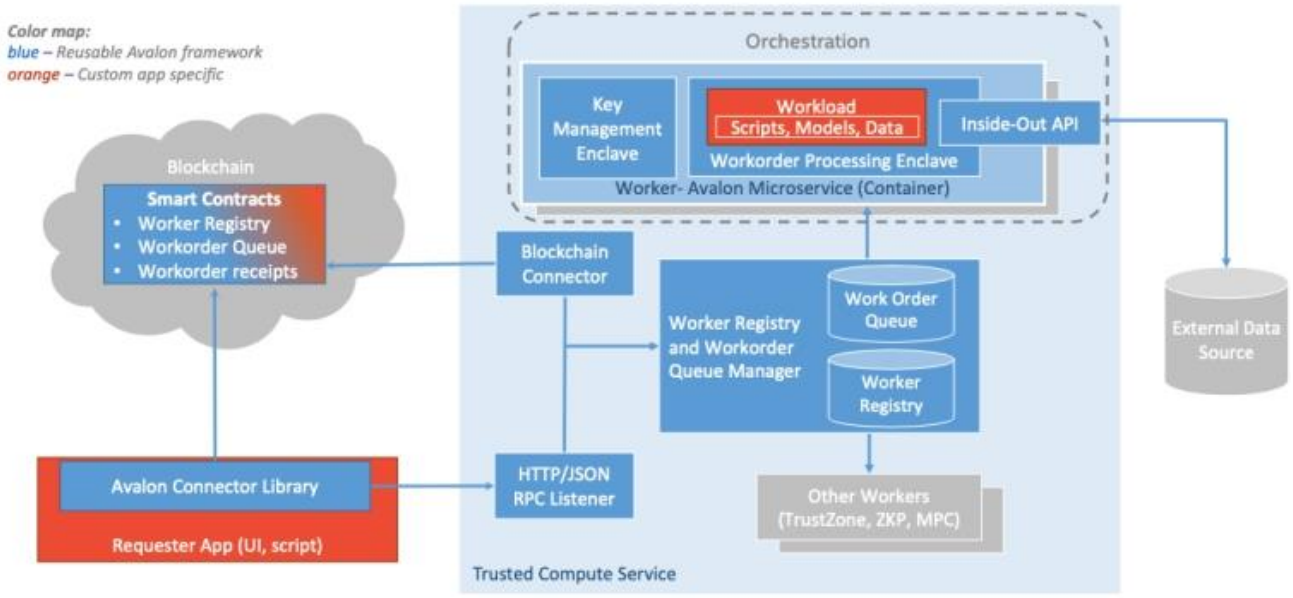


Рисунок 2.15 – Архітектура HLA [43]

Структура HLA вимагає безліч компонентів для роботи і таким чином, налаштування мережі з використанням HLA є досить складним процесом[43]. У новій схемі пропонується простіше рішення – реалізація верифікації коду в TEE за допомогою смарт-контракту перед кожним новим виконанням цього коду. Такий підхід полегшує взаємодію компонентів і забезпечує більш зручний аудит системи, при цьому для бізнес-компаній рішення залишається автоматизованим, що скорочує економічні витрати та економить час.

2.5 Висновок

Таким чином, у даному розділі випускної кваліфікаційної роботи було розглянуто найпопулярніші blockchain платформи серед яких обрали найперспективнішу та найбільш розвинену для нашої роботи. Також було обрано мову програмування, на якій було написано смарт-контракт, за допомогою якого було підвищено достовірність системи електронного голосування. Розглянули механізм довіреного середовища виконання з прикладу Intel SGX. Були описані особливості реалізації додатків для Intel SGX із застосуванням анклавів. У створенні довіреного середовища для обробки персональних даних при голосуванні було обрано технологію довіреного

середовища виконання Intel SGX за рахунок найбільш відповідного механізму «сліпої» обробки конфіденційних даних для даної роботи. Застосування анклавів дозволить передавати персональні дані користувачів на довірене середовище з упевненістю в безпеці цих даних.

ПРОГРАМНА РЕАЛІЗАІЯ ТА ТЕСТУВАННЯ РОЗРОБЛЕНОЇ СИСТЕМИ ГОЛОСУВАННЯ

3.1 Тестування смарт-контракту

Для того щоб здійснити проміжну перевірку роботи смарт контракту застосовується інструмент Ganache-CLI, який запускається з bash-консолі та створює локальний тестовий блокчейн із десятьма акаунтами. Викликається він із консолі nodejs, командою ganache-cli, після цього нам показується 10 тестових облікових записів. Тестова мережа розміщується за адресою localhost:8545.

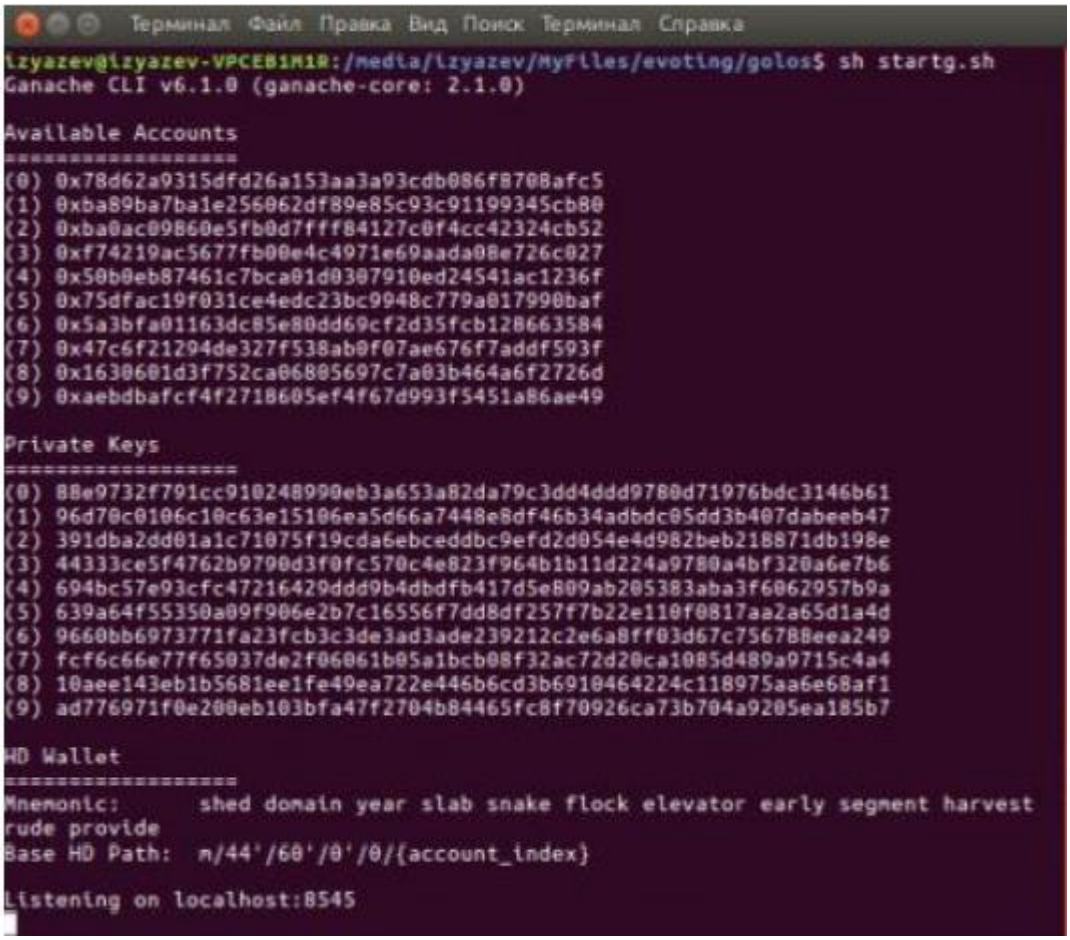


Рисунок 3.1 – тестова blockchain мережа для перевірки роботи смарт контракту

Після того, як тестова блокчейн мережа піднята, виконується компіляція смарт-контракту з розширенням ".sol" в інтегрованому середовищі розробки Remix. Далі необхідно перейти у вкладку «Run» у середовищі розробки та в полі «Environment» вказати адресу створеної за допомогою ganache-cli web3-

провайдера, тобто http://localhost:8545. Після цього в полі «Account» відобразяться існуючі в мережі тестові облікові записи, як на малюнку 3.2.

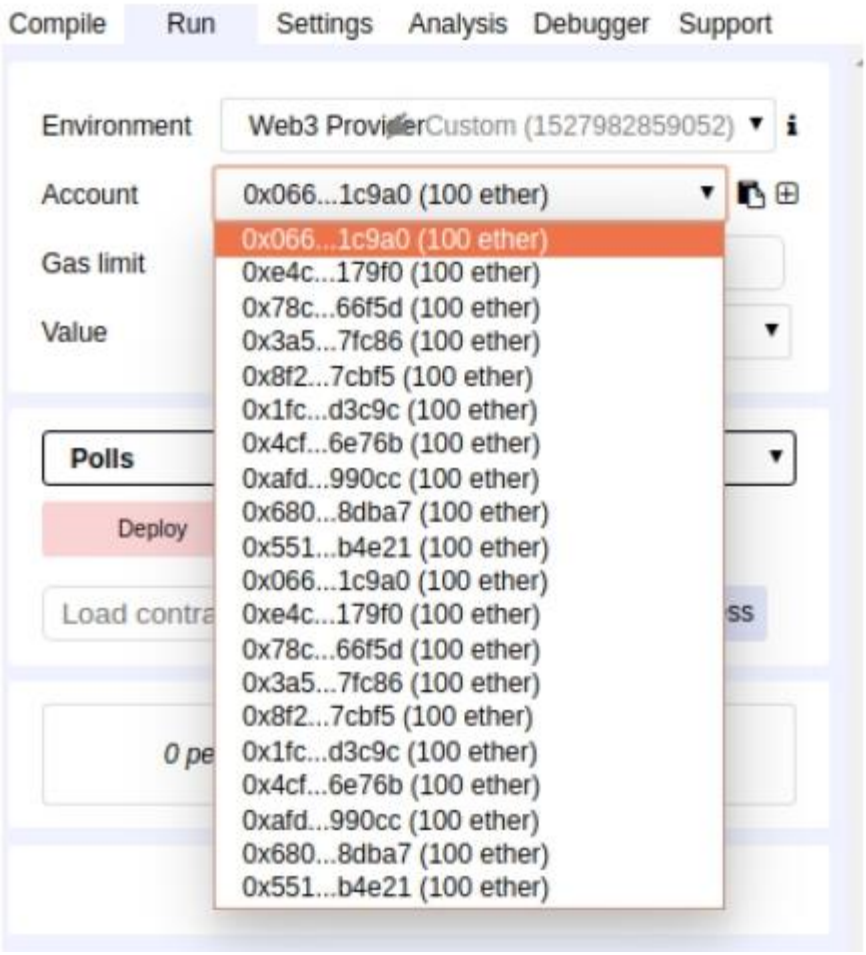


Рисунок 3.2 – Підключення Remix до тестової мережі ganache-CLI

Після цих маніпуляцій потрібно розгорнути контракт за допомогою кнопки «Deploy» та протестувати його роботу, починаючи з наповнення конструктора інформацією про опитування, що створюється.

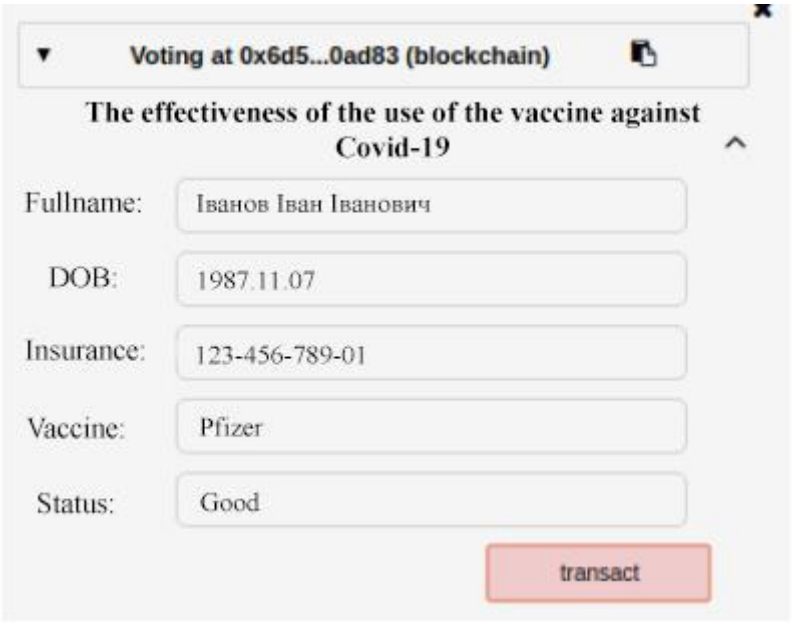


Рисунок 3.3 – Наповнення конструктора вхідними даними

Далі слід протестувати функцію голосування.

Подивитися, чи успішно спрацювала функція можна у ganache-cli. На малюнку 3.4 показано, що після того, як ми задіяли функцію голосування, успішно створилася транзакція та сформувалася в блок №3(блок №1 сформувався при розгортанні транзакції, а блок №2 при створення опитування).

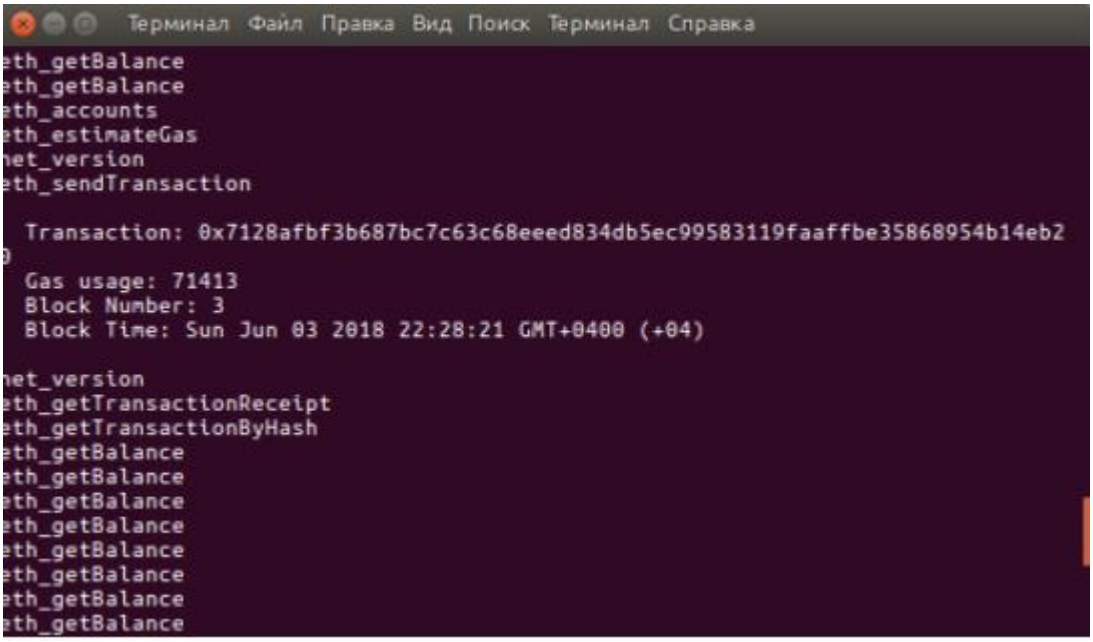


Рисунок 3.4 – Сформований блок підтверджує успішне виконання функції

Після тестування основних функцій слід додати деякі обмеження голосування. Так, на даний момент до такої системи може приєднатися будь-

який учасник та проголосувати. Щоб цього уникнути, необхідно продумати як вирішити цю проблему.

Реєстрація передбачена білим списком. У цьому смарт контракті реалізовано можливість створювати необмежену кількість голосувань. Це означає, що такою системою можуть користуватися різні організації, різні люди можуть створювати своє голосування, своє обговорюване опитування та варіанти відповідей. У системі, що розробляється голосування було вирішено призначити кожному опитуванню адміністратора, який буде творцем опитування. Після того, як користувач створює опитування, йому надається статус creator, користуючись яким він може викликати функцію створення білого списку та додати туди необхідні акаунти в мережі блокчейн, яким буде дозволено голосувати.

```
function createVoting (string _question, string _name, string _option1, st

    polls[_name].title = _name;
    polls[_name].question = _question;
    polls[_name].option1 = _option1;
    polls[_name].option2 = _option2;
    polls[_name].option3 = _option3;
    polls[_name].count1 = 0;
    polls[_name].count2 = 0;
    polls[_name].count3 = 0;
    polls[_name].exists = true;
    creator = msg.sender;
}
```

Рисунок 3.5 – Надання творцю опитування статусу creator

```
function addWhitelistAddress (address _address) public {
    if (msg.sender != creator) return;
    whitelist[_address] = true;
}

function addManyWhitelistAddress (address[] _addresses) public {
    if (msg.sender != creator) return;
    for (uint256 i = 0; i < _addresses.length; i++) {
    whitelist[_addresses[i]] = true;
   }
}
```

Рисунок 3.6 – Функція білого списку

При виконанні функція білого аркуша спочатку відбувається перевірка наявності у користувача статусу creator.

Тепер, щоб брати участь у голосуванні, користувачеві потрібно мати обліковий запис у приватній або публічній мережі blockchain, в якій розгорнуть

цей смарт-контракт та дозвіл від творця, який надається при внесенні користувача до білого списку.

Розробка смарт-контракту закінчено.

3.2 Розгортання смарт-контракту на реальну blockchain-мережу

Після того, як розроблено смарт контракт і web-інтерфейс, щоб система голосування працювала як потрібно, необхідно розгорнути смарт контракт у реальній блок-кайн-мережі. Так як у blockchain-симуляції ganache CLI смарт-контракт був успішно протестований, то для завершення проекту залишилося виконати лише кілька маніпуляцій.

Для початку потрібно встановити клієнт Go Ethereum з репозиторію Ethereum. Робиться це за допомогою таких команд, як:

- apt-get install software-properties-common.
- add-apt-repository -y ppa:ethereum/ethereum.
- apt-get install ethereum.

Розгорнути смарт-контракт можна або на приватній мережі мережі Ethereum, або на публічній.

Приватний Ethereum-blockchain можна створити засобами клієнта geth, встановленого раніше. Така мережа буде закрита від усього решти світу та надана тільки розробнику, який її запустив і тим, кому він дав до неї доступ.

Перевагою приватної мережі є швидка швидкість транзакцій, так як така мережа не навантажена. Однак така мережа матиме низку мінусів. Так, наприклад, у разі цього проекту електронного голосування, зважаючи на те, що через невелику кількість користувачів, а на даний момент учасників системи голосування взагалі немає, не буде кому обробляти транзакції та верефікувати блоки. До того ж, при невеликій кількості користувачів суттєво збільшується ризик «Атаки 51», при якому більше 50% майнерів можуть об'єднатися та маніпулювати даними в блокчейні на власний розсуд.

Публічною мережею є, наприклад, основна мережа Ethereum, яка запустилася у 2015 році[44], у день презентації платформи. Також, є велика

кількість публічних мереж Ethereum, але не є офіційними мережами проекту. У таких мережах «Атака 51» практично неможлива через велику кількість майнерів та учасників мережі. У публічної мережі на таку атаку доведеться витратити більше коштів, ніж можна отримати.

Розгортати смарт-контракт планується в Ethereum-мережі з назвою Rinkeby. Вибір зроблено у бік неофіційної блокчейн мережі Ethereum, так як при розгортанні договору необхідно завантажити всю базу даних мережі, а це дуже ресурсомістка задача. Знадобиться значне кількість часу та пам'яті на диску, так як офіційна blockchain Ethereum станом на 2022 рік має у своїй базі даних близько 500 Гб[45].

Для того, щоб запустити на своєму пристрої вузол Ethereum Rinkeby, необхідно виконати команду ^

geth --rinkeby --syncmode "fast" --rpc --rpcapi db,eth,net,web3,personal --cache=1024 --rpcport 8545 --rpcaddr 127.0.0.1 --rpccorsdomain "*".

Після цього почнеться завантаження блокчейна, в середньому дана операція займає близько 1:00.

Після того, як блокчейн завантажився, для розгортання смарт контракту знадобиться фреймворк Truffle, який дозволить розгорнути договір у публічній мережі.

Далі необхідно розпакувати Truffle у директорії з усіма файлами, які були створені у процесі розробки системи голосування, упорядкувати файли необхідним фреймворком.

Після того, як смарт контракт і засоби для взаємодії з системою готові, потрібно створити обліковий запис за допомогою команди в консолі node.js web3.personal.newAccount('xxx'), де xxx – пароль від створюваного облікового запису.

Потім необхідно, щоб у облікового запису, який буде голосувати на балансі було кілька токенів, щоб за допомогою них здійснювати голосування, яке відбувається у вигляді транзакції. Для цього слід або купити їх за реальні гроші, або скористатися сервісами, які раз в якусь кількість часу переводять на

потрібний обліковий запис символічну суму на знак подяки за користування їх проектом.

Після поповнення балансу облікового запису виборця потрібно розблокувати аккаунт, так як за замовчуванням він заблокований, робиться це за допомогою команди

web3.personal.unlockAccount('0xds651g6s5g6d654684654x3g354352srg654g', 'xxx', 13782), де після команди в дужках йде хеш акаунту, пароль та кількість "палива" для виконання операції.

Після всіх цих маніпуляцій виконується компіляція та розгортання смарт-контракту, яка відбувається так само, як і в тестовій мережі, і після невеликої кількості часу, його статус змінюється на стан готовності до роботи.

Для подальшої взаємодії користувача з контрактом зараз необхідно в IDE Remix перейти до розділу «Details» і скопіювати всю інформацію з поля «ABI», яка показана на малюнку 3.7:

Рисунок 3.7 – ABI інтерфейс

Ця маніпуляція необхідна для того, щоб у майбутньому користувач міг використовувати графічний інтерфейс для роботи зі смарт контрактом.

Крім цього, у консолі Truffle необхідно ввести deployedContract.adress, тобто адреса смарт-контракту в мережі blockchain, це також необхідно для подальшої взаємодії через графічний інтерфейс.

3.3 Графічний інтерфейс взаємодії із захищенним голосуванням

Взаємодія із системою голосування здійснюватиметься за допомогою ПЗ Mist від розробників проекту Ethereum. Дане ПЗ є браузером і дозволяє взаємодіяти користувачеві зі смарт-контрактом за допомогою свого графічного інтерфейсу.

Після того, як раніше, в мережі rinkeby був розгорнутий смарт-контракт, користувачеві необхідно встановити собі на комп'ютер ПЗ Mist з офіційного репозиторію, який знаходиться за адресою https://github.com/ethereum/mist/releases, це ПЗ підтримується практично на всіх операційних системах.

Після цього, йому необхідно створити обліковий запис, зробити це можна натиснувши кнопку «Додати обліковий запис».

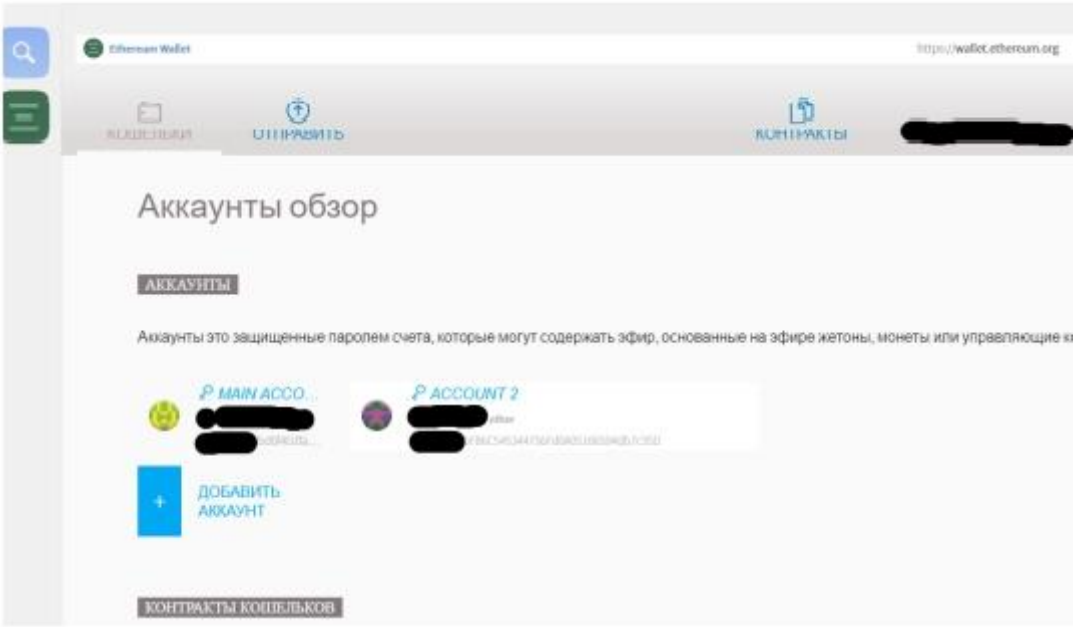


Рисунок 3.8 – Створення облікового запису для подальшої взаємодії зі смарт-контрактом

Після цього користувачеві необхідно перейти у вкладку "Контракти", і натиснути на кнопку "Спостерігати за контрактом". Відкриється вікно з полями для введення «Адреса контракту», «Назва контракту» та «JSON інтерфейс». Користувачеві необхідно заповнити ці поля. У нашому випадку, адреса розгорнутого контракту "0x4638D059097Ee13d97aC51919519549E844cC473", назва контракту "Voting", а JSON інтерфейс це ABI, яке було збережено з середовища розробки Remix.

Після цього у користувача з'являється можливість використовувати все функції контракту, розроблені.

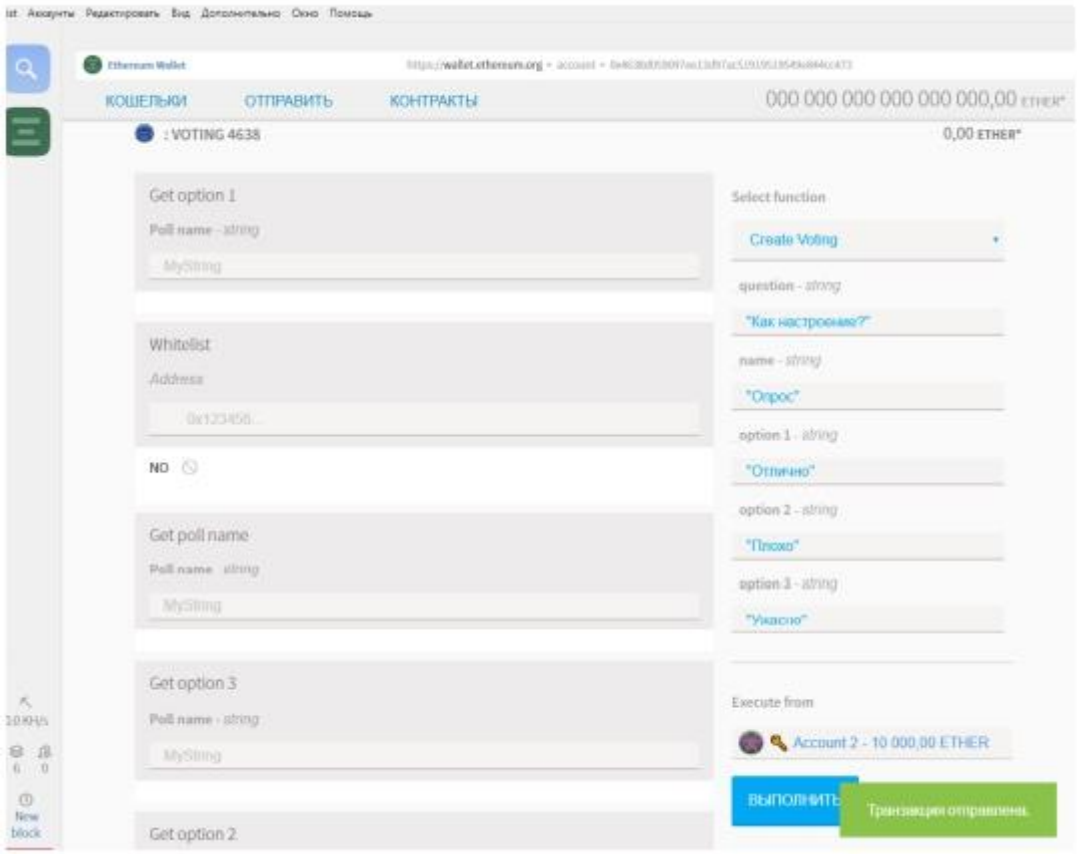


Рисунок 3.9 – Взаємодія зі смарт-контрактом через браузер Mist

3.4 Виявлення недоліків та способи їх усунення

. З недоліків можна відзначити низьку швидкість роботи додатків з анклавами Intel SGX [46]. Група дослідників провела експеримент на двох машин, оснащених процесором Intel Core i7-7700 з підтримкою Intel SGX (чотирьохядерний з SMT; 3,60 ГГц) та 32 ГБ оперативної пам'яті, що працює під

керуванням Ubuntu 18.04.3 LTS та підключеними через гігабітний Ethernet. Вибіркове дослідження складалося з короткого демографічного опитувальника із 10 пунктів.

Дослідники послідовно відправляли на платформу до 10 000 учасників для кожного прогону оцінки перед переходом до виконання статистичного аналізу. Для кожного надсилання відповіді на опитування вимірювався загальний час відповіді за клієнта. Для кожного запуску дослідження також вимірювали загальний час виконання аналізу після збирання всіх відповідей. Реалізація без анклав зайняла в середньому 15,8 мс для обробки. У свою чергу, для обробки в анклаві потрібно 297 мс. На малюнку 4.2 показано вплив різних розмірів вибірки (від 10 до 1000) на час виконання із застосуванням SGX і без нього. Для розміру вибірки! = 10 000 середній час обчислень з використанням SGX у 15,7 разів більше, ніж без SGX (4,6 с та 72,4 с).

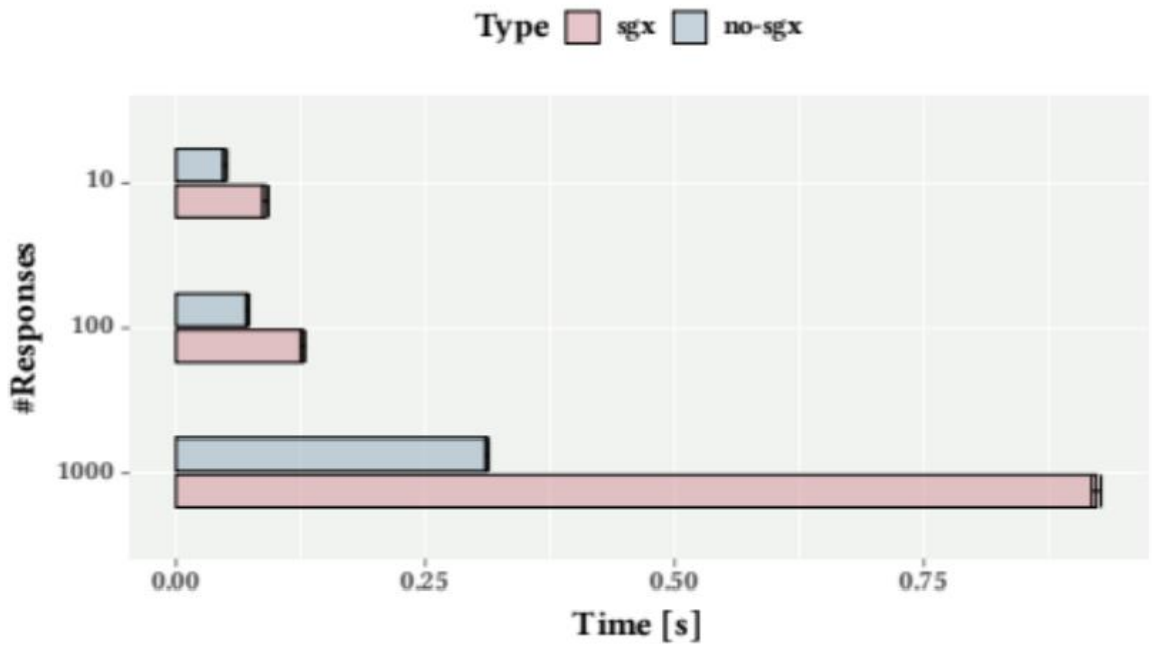


Рисунок 3.10 – Порівняння часу роботи з анклавом та без нього

Для усунення цієї проблеми слід скорочувати код анклаву до мінімуму, залишаючи найнеобхідніші функції і додавати паралельні анклави, щоб розподіляти додаткове навантаження.

Ще одним недоліком може бути те, що використання Intel SGX накладає обмеження на використовувану техніку. Дане розширення підтримують всі

сучасні процесори Intel, починаючи з 5го покоління, на більш старі версії використовувати Intel SGX не вийде. Рішенням може бути запуск SGX на віртуальній машині в режимі симуляції або альтернативне рішення з ARM TrustZone.

Список процесорів Intel, схильних до атак CrossTalk представлений на малюнку 3.11.

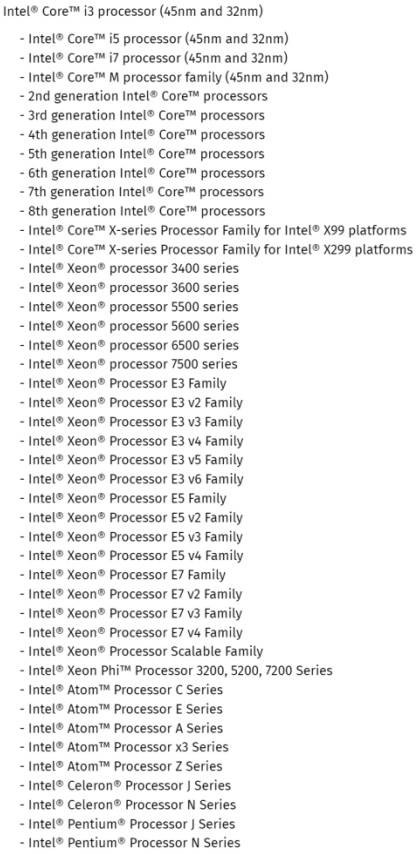


Рисунок 3.11 – Вразливі до атаки CrossTalk процесори Intel

Компанія Intel випустила оновлення мікрокоду, яке виправляє цей баг. Оновлення блокує всю шину пам'яті перед оновленням staging buffer (проміжний буфер), а розблокує її тільки після очищення вмісту. Intel застосовує зміни лише до особливо важливих з погляду безпеки інструкцій, включаючи RDRAND, RDSEED. Дані іншої інструкції можна так само прочитати з буфера.

Поява нових side-channel attacks створює тенденцію до постійного оновленню мікрокоду процесорів Intel, що позначається на роботі самих процесорів, іноді уповільнюючи їх роботу. Рішенням може стати перехід

компанії Intel на новий тип процесорів, що застосовують більш надійні методи захисту від атак сторонніми каналами.

3.5 Висновок

Таким чином, у даному розділі випускної кваліфікаційної роботи розроблене рішення було протестовано у мережі із підключеним зовнішнім захищенним середовищем із анклавом Intel SGX. Перш за все було проведенно тестування розробленої системи в локальній тестовій мережі, після чого розгорнули написаний смарт-контракт в реальну blockchain мережу. Описані маніпуляції необхідні були для подальшої успішної взаємодії через графічний інтерфейс. Також були виявлені недоліки та способи їх вирішення.

Розроблена система голосування відповідає всім вимогам безпеки, щоб ефективно використовувати в різних сферах. Використання блокчейн в даному рішенні дозволяє об'єднувати дані різних незалежних компаній в одну систему і виробляти точні обчислення.

ЕКОНОМІЧНА ЧАСТИНА

1.1 Оцінювання комерційного потенціалу розробки програмного забезпечення

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Результатом магістерської кваліфікаційної роботи є підвищення достовірності системи голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів.

Для проведення технологічного аудиту залучено трьох незалежних експертів. У межах даної роботи такими експертами є викладачі кафедри МБІС: Карпінець В. В. (к.т.н., доцент каф. МБІС ВНТУ), Салієва О.В. (д.ф., викл. каф. МБІС ВНТУ) та Шиян А. А. (к.ф.-м.н., доцент каф. МБІС ВНТУ).

Оцінювання комерційного потенціалу здійснимо за критеріями, що наведені в таблиці 4.1

Таблиця 4.1 – Критерії оцінювання комерційного потенціалу розробки бальна оцінка.

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експл. витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навч. наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження таблиці 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої к-ті дозвільних документів на вир-во та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу експертами розробки зведено в таблицю 4.2.

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	1 – Карпінець В.В.	2 – Салієва О.В.	3 – Шиян А.А
1	3	4	3
Ринкові переваги (недоліки):			
2	3	3	4
3	4	3	3
4	3	4	3
5	3	3	4
Ринкові перспективи			
6	4	4	3
7	3	3	3
Практична здійсненність			
8	4	4	4
9	3	3	3
10	3	3	3
11	3	4	3
12	4	4	3
Сума балів	$СБ_1 = 40$	$СБ_1 = 42$	$СБ_1 = 39$
Середньоарифметична СБ	$\overline{СБ} = 40{,}3$		

За даними таблиці 4.2 можна зробити висновок, щодо рівня комерційного потенціалу розробки. Зважимо на результат й порівняємо його з рівнями комерційного потенціалу розробки, що представлено в таблиці 4.3.

Таблиця 4.3 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 48	Високий

Рівень комерційного потенціалу розробки, становить 40,3 балів, що відповідає рівню «вище середнього».

Проаналізуємо суть технічної проблеми та розглянемо аналоги. Наукова новизна розробки полягає в:

- створенні механізму верифікації коду анклаву Intel SGX за допомогою смарт-контракту;

- надання можливості обробки даних без порушення їх конфіденційності.

Розробка програмного засобу має на меті практичну реалізацію удосконаленого алгоритму електронної системи голосування в мережі блокчейн на основі смарт-контрактів.

Застосування програмного електронного ключа для поставленої задачі зумовлено такими його перевагами як зниження витрат, підвищення зручності користування, зниження ймовірності втрати чи крадіжки та менший ризик атак через посередника.

Під час виконання роботи було здійснено тестування розробленого прорамного продукту, яке показало позитивні результати, показники яких свідчать про можливість застосування розробленої системи голосування на практиці.

Програмний засіб на сьогодні має перспективу та користь як для

пересічних користувачів так і для комерційних служб. Продукт, який пропонується є реалізованим засобом, що дозволяє проводити електронне голосування в мережі блокчейн.

Для реалізованого проекту є характерним можливість надання якісних послуг з технічної підтримки на всіх етапах використання програмного продукту користувачами.

Передбачається, що засобами розповсюдження та збуту розробки можуть бути: інтернет-магазини, фінансова сфера, медична та всі ті, де використовуються конфіденційні персональні дані.

Доцільним є встановлення середньої ціни на програмний продукт задля якісного та швидкого виходу на ринок послуг. Також це може допомогти, як в боротьбі з конкурентами, так і в захопленні більшої частки ринку.

Також передбачається проведення рекламної компанії для залучення інвесторів та поширення інформації про продукт серед пересічних користувачів. Просування продукції на ринку планується здійснювати завдяки цільовій рекламі.

1.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів

Прогнозування витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи складається з таких етапів:

1-й етап: розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи;

2-й етап: розрахунок загальних витрат на виконання даної роботи;

3-й етап: прогнозування загальних витрат на виконання та впровадження результатів даної роботи.

Виконаємо розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи, за такими статтями та формулами, приймаючи до уваги

те, що для розробки інформаційної технології було залучено одного розробника програмного забезпечення.

1. Основна заробітна $З_о$:

$$З_о = \frac{М}{Т_р} \cdot t, \text{грн.} \quad (4.1)$$

де М – місячний посадовий оклад – 15 000 грн.;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р = 22$ дні;

t – число робочих днів роботи – 30 днів.

Таким чином:

$$З_о = \frac{15000}{22} \cdot 30 = 20\,450 \text{ (грн.)}$$

Таблиця 4.4 – Витрати по заробітній платі

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату
Розробник	15 000	680	30	20 400
Всього				20 400

2. Додаткова заробітна плата $З_д$ працівників розраховується як 12% від основної заробітної плати:

$$З_д = 0{,}12 \cdot 20\,400 = 2\,450 \text{(грн.)} - \text{для розробника}$$

3. Нарахування на заробітну плату $Н_{зп}$ розробника становить:

$$Н_{зп} = (З_о + З_д) \cdot \frac{\beta}{100} \quad (4.2)$$

де $З_о$ – основна заробітна плата розробника;

$З_д$ – додаткова заробітна плата розробника;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування – 22%.

$$Н_{зп} = (20\,400 + 2\,450) \cdot 0{,}22 = 5\,027 \text{(грн.)}$$

4. Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час виконання даного етапу роботи розраховуємо за формулою:

$$А = \frac{Ц \cdot Т}{12 \cdot Т_В} \quad (4.3)$$

де Ц – загальна балансова вартість обладнання, приміщення тощо, грн.;

Т – фактична тривалість використання, міс;

$Т_В$ – термін використання обладнання, приміщень тощо, роки.

Розробка програмного забезпечення ведеться орієнтовно 1,5 місяці.

Для офісного приміщення $А = \frac{120\,000 \cdot 1,5}{12 \cdot 5} = 3\,000грн.$; для комп'ютера $А = \frac{15\,000 \cdot 1,5}{12 \cdot 2} = 937,5грн.$; для монітора $А = \frac{8\,000 \cdot 1,5}{12 \cdot 2} = 500грн.$

Розрахунки зведено до таблиці 4.6:

Таблиця 4.5 – Амортизаційні відрахування

Наймену-вання	Балансова вартість (грн.)	Термін використання (років)	Фактична тривалість в-ня, (міс.)	Величина ам.. відрахувань, (грн..)
Офісне приміщення	120 000	5	1,5	3 000
Комп'ютер	15 000	2	1,5	937,5
Монітор	8 000	2	1,5	500
			Всього	4 437,5

5. Витрати на комплектуючі К, що були використані під час виконання даного етапу роботи, розраховуються за формулою:

$$К = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i \ (грн.) \quad (4.4)$$

де $Н_i$ – кількість комплектуючих *i*-го виду, шт.;

$Ц_i$– ціна комплектуючих *i*-го виду, грн.;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

n – кількість видів комплектуючих.

Таблиця 4.6 – Витрати на комплектуючі

Найменування комплектувальних	Кількість	Ціна за штуку, грн.	Сума, грн.	Примітка
Клавіатура	1	750 грн.	750 грн.	
Комп'ютерна мишка	1	350 грн.	350 грн.	
Всього:			$К_і = 1,2$	1 100 грн.

6. Витрати на силову електроенергію $В_е$ розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п \text{ (грн.)} \quad (4.5)$$

де В – вартість 1 кВт-год. (на сьогодні для підприємців вартість 3,45грн./кВт-год);

П – установлена потужність обладнання – 0,8 кВт;

Ф – фактична кількість годин роботи обладнання – 350 годин,

$К_п$ – коефіцієнт використання потужності.

$$В_е = 6,4 \cdot 0,8 \cdot 350 \cdot 0,14 = 250,9 \text{ (грн).}$$

7. Інші витрати $В_{ін}$ охоплюють:

– витрати на управління організацією;

– оплату службових відряджень;

– витрати на утримання, ремонт та експлуатацію основних засобів;

– витрати на опалення, освітлення, водопостачання, охорону праці тощо.

Інші витрати $В_{ін}$ можна прийняти як 100% від суми основної заробітної плати розробника:

$$В_{ін} = 31\,500 \cdot 1 = 31\,500 \text{ (грн)}$$

Послуги Інтернету – 350 грн., канцтовари – 300 грн. Загальна вартість становить:

$$350 + 300 = 650 \text{ (грн.)}$$

8. Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи – В.

$$В = 20\,450 + 2\,450 + 5\,027 + 4\,437,5 + 1100 + 250,9 + 20\,400 + 650 = 54\,765 \text{ (грн.)}$$

9. Проведемо прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи. Прогнозування здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta}, \text{грн.} \quad (4.6)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Так, якщо розробка знаходиться:

– на стадії науково-дослідних робіт, то $\beta \approx 0{,}1$;

– на стадії технічного проектування, то $\beta \approx 0{,}2$;

– на стадії розробки конструкторської документації, то $\beta \approx 0{,}3$;

– на стадії розробки технологій, то $\beta \approx 0{,}4$;

– на стадії розробки дослідного зразка, то $\beta \approx 0{,}5$;

– на стадії розробки промислового зразка, $\beta \approx 0{,}7$;

– на стадії впровадження, то $\beta \approx 0{,}9$.

$В_{заг}$ – загальна вартість всієї наукової роботи.

$$В = 54\,765 \ (\text{грн.})$$

$$ЗВ = \frac{54\,765}{0{,}7} = 78\,236 \ (\text{грн.})$$

Отже, прогноз загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи складає $78\,236$ (грн.)

1.3 Прогнозування комерційних ефектів від реалізації результатів розробки

У даному підрозділі проведемо кількісне прогнозування, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи.

В умовах ринку узагальнюючим позитивним результатом, що його отримує підприємство від впровадження результатів тієї чи іншої розробки, є

збільшення чистого прибутку підприємства. Зростання чистого прибутку можна оцінити у теперішній вартості грошей.

Зростання чистого прибутку забезпечить інвестору надходження додаткових коштів, які дозволять покращити фінансові результати діяльності.

Виконання даної наукової роботи та впровадження її результатів складає приблизно 1 рік. Позитивні результати від впровадження розробки очікуються вже в перші місяці після впровадження.

Проведемо детальне прогнозування позитивних результатів та кількісне їх оцінювання по роках.

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

$$\Delta П_i = \sum_{1}^{n} (\Delta П_я \cdot N + П_я \cdot \Delta N)_i \qquad (4.7)$$

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.

Припустимо, що внаслідок впровадження результатів наукової розробки чистий прибуток підприємства збільшиться на 180 грн., а кількість одиниць реалізованої послуги збільшиться:

– протягом першого року – на 450 од.,

– протягом другого року – ще на 600 од.,

– протягом третього року – ще на 800 од.

Орієнтовно: реалізація продукції до впровадження результатів наукової розробки складала 1 шт., а прибуток, що його отримувало підприємство на одиницю продукції до впровадження результатів наукової розробки – 150 грн.

Потрібно спрогнозувати збільшення чистого прибутку підприємства від впровадження результатів наукової розробки у кожному році відносно базового.

Збільшення чистого прибутку підприємства $\Delta П_1$ протягом першого року складе:

$$\Delta П_1 = 150 \cdot 1 + (150 + 180) \cdot 450 = 148\,650 \text{ (грн.)}$$

Обчислимо збільшення чистого прибутку підприємства $\Delta П_2$ протягом другого року:

$$\Delta П_2 = 150 \cdot 1 + (150 + 180) \cdot (450 + 600) = 346\,650 \text{ (грн.)}$$

Збільшення чистого прибутку підприємства $\Delta П_3$ протягом третього року становитиме:

$$\Delta П_3 = 150 \cdot 1 + (150 + 180) \cdot (450 + 600 + 800) = 610\,650 \text{ (грн.)}$$

Отже, розрахунки показують, що відповідно прогнозуванню комерційний ефект від впровадження розробки виражається у значному збільшенні чистого прибутку підприємства.

1.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Основними показниками, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахунок ефективності вкладених інвестицій передбачає:

1-й крок. Розрахунок теперішньої вартості інвестицій PV, що вкладаються в наукову розробку.

Такою вартістю ми можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, тобто $ЗВ = PV = 115\,523{,}3$ (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:

$$\Delta П_1 = 121\,650\ (\text{грн}), \Delta П_2 = 283\,650\ (\text{грн}), \Delta П_3 = 610\,650 (\text{грн}).$$

3-й крок. Будуємо вісь часу, на якій відображаємо всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів. Рисунок 4.1 характеризує рух платежів (інвестицій та додаткових прибутків).

Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4-й крок. Розрахуємо абсолютну ефективність вкладених інвестицій $Е_{абс}$ за формулою:

$$Е_{абс} = (ПП - PV), (\text{грн.}) \qquad (4.8)$$

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

PV – теперішня вартість інвестицій $PV = ЗВ, \text{грн.}$

Приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, (грн) \quad (4.9)$$

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 – 0,1;

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

$$ПП = \frac{148\,650}{(1+0{,}1)^1} + \frac{346\,650}{(1+0{,}1)^2} + \frac{610\,650}{(1+0{,}1)^3} = 880\,623\ (грн.)$$

$$Е_{абс} = 880\,623 - 115\,685{,}7 = 764\,938\ (грн.)$$

Оскільки $Е_{абс} > 0$, результат від проведення наукових досліджень щодо розробки програмного продукту та їх впровадження принесе прибуток, тобто є доцільним, проте це ще не свідчить про те, що інвестор буде зацікавлений у фінансуванні даної програми.

5-й крок. Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$ за формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1 \quad (4.10)$$

де $Е_{абс}$ – абсолютна ефективність вкладених інвестицій, грн.;

PV – теперішня вартість інвестицій $PV = ЗВ, грн.$

$Т_ж$ – життєвий цикл наукової розробки, роки.

$$E_в = \sqrt[3]{1 + \frac{764\,938}{115\,685{,}7}} - 1 = \sqrt[3]{6{,}6} - 1 = 0{,}79 \text{ або } 79\%$$

Порівняємо $E_в$ з мінімальною (бар'єрною) ставкою дисконтування τ_{min}, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. Спрогнозуємо величину τ_{min}.

У загальному вигляді мінімальна (бар'єрна) ставка дисконтування τ_{min}визначається за формулою:

$$\tau_{min} = d + f \tag{4.11}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; $d = 0{,}2$; f – показник, що характеризує ризикованість вкладень; величина $f = 0{,}5$.

$$\tau_{min} = 0{,}2 + 0{,}5 = 0{,}7$$

Оскільки $E_в = 79\% > \tau_{min} = 70\%$, то у інвестора є потенційна зацікавленість у фінансуванні даної наукової розробки.

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

$$Т_{ок} = \frac{1}{Е_в}, \text{рік} \tag{4.12}$$

$$Т_{ок} = \frac{1}{0{,}79} = 1{,}26 \text{ (року)}$$

Оскільки термін окупності вкладених у реалізацію наукового проекту інвестицій менше трьох років ($Т_{ок} < 3$ років), то фінансування нової розробки є доцільним.

1.5 Висновки до розділу

В даному розділі було виконано оцінювання комерційного потенціалу розробки системи електронного голосування в мережі блокчейн на основі смарт-контрактів.

Проведено технологічний аудит з залученням трьох незалежних експертів. Визначено, що рівень комерційного потенціалу розробки вище середнього. Згідно з проведеним оцінюванням нова розробка є якісною та конкурентоспроможною.

Рівень комерційного потенціалу розробки, становить 40,3 балів, що відповідає рівню «вище середнього».

Згідно із розрахунками всіх статей витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи загальні витрати на розробку складають 115 685,7 (грн.). Розрахована абсолютна ефективність вкладених інвестицій в сумі 764 938 (грн.) свідчить про отримання прибутку інвестором від комерціалізації програмного продукту.

Щорічна ефективність вкладених в наукову розробку інвестицій складає 79%, що вище за мінімальну бар'єрну ставку дисконтування, яка складає 70%. Це означає потенційну зацікавленість інвесторів у фінансуванні розробки.

Термін окупності вкладених у реалізацію проекту інвестицій становить 1,26 (року), що також свідчить про доцільність фінансування нової розробки.

Отже, проаналізувавши отримані економічні показники, можна вважати, що запропонована розробка програмного засобу має високий комерційний потенціал, а тому є доцільною для подальшого впровадження.

ВИСНОВКИ

Метою даної випускної кваліфікаційної роботи було підвищення достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів. У розробленій системі реалізовано функції голосування, реєстрація в системі реалізована шляхом створення облікового запису в мережі blockchain, в якій розгорнуть смарт-контракт та додаванням користувача у білий список того чи іншого створеного опитування.

У роботі було досліджено сучасні механізми, що надають можливість довірених віддалених обчислень та розглянуто підходи до побудови децентралізованих мереж блокчейн. Об'єднання цих двох механізмів доповнило один одного і дозволило розширити кількість підходів до збільшення довіри до системи.

Розроблена система голосування працює в blockchain мережі Ethereum та використовує створення анклавів Intel SGX для ізоляції персональних даних від ненадійної частини ОС. Використання надійних алгоритмів шифрування та схеми підпису забезпечує конфіденційність і цілісність критично важливих даних.

У ході виконання роботи було вивчено теоретичний базис з технології blockchain, платформи для розробки децентралізованих додатків Ethereum, мова програмування Solidity, якою відбувається розробка смарт-контрактів, що регулюють умови для маніпулювання даними у мережі blockchain.

Проводився аналіз існуючих рішень систем електронного голосування, виявлено їх недоліки порівняно з аналогічною системою, реалізованою за допомогою технології blockchain. Крім того, були проаналізовано різноманітні засоби розробки децентралізованої програми на платформі Ethereum, інтегроване середовище розробки Remix, програмна платформа node.js, спеціальні бібліотеки node.js, протокол Ethereum go, що дозволяє запустити приватну мережу блокчейн або підключитися до існуючої.

Розробка та тестування децентралізованого додатку здійснювалося на двох пристроях із операційними системами Ubuntu 16.04 та Windows 10.

Розроблений децентралізований додаток дозволяє запускати голосування з бажаними об'єктами голосування, реєструвати облікові записи виборцям для користування системою. Програма має простий і зрозумілий користувачеві інтерфейс, реалізований ПЗ Mist, що дозволяє зручно працювати із функціями смарт-контракту.

Завдання, поставлені на початку роботи були виконані, мета випускної кваліфікаційної роботи була досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Block The Vote: Could Blockchain Technology Cybersecure Elections? [Електронний ресурс]. – Режим доступу: http://www.forbes.com/sites/realspin/2016/08/30/block-the-votecouldblockchain-technology-cybersecure-elections.

2. California: The Top to Bottom Review [Електронний ресурс]. – Режим доступу: http://www.votetrustusa.org/index.php?option=com_content&task=view&id=2554&Itemid=113.

3. IGS Votomatic Prototype Goes to the Smithsonian [Електронний ресурс]. – Режим доступу: https://web.archive.org/web/20070713201451/http://www.igs.berkeley.edu/publicatios/par/winter2001/votomatic.htm.

4. Kiwi. Bitcoin testnet sandbox. [Електронний ресурс]. – Режим доступу: https://testnet.manu.backend.hamburg/faucet.

5. NSW election result could be challenged over iVote security flaw [Електронний ресурс]. – Режим доступу: https://www.theguardian.com/australia-news/2015/mar/23/nsw-electionresult-could-be-challenged-over-ivote-security-flaw.

6. Peer-to-peer [Електронний ресурс]. – Режим доступу: https://bitcoin.org/bitcoin.pdf.

7. Slim. Middeware-slim. [Електронний ресурс]. – Режим доступу: https://www.slimframework.com/docs/v3/concepts/middleware.html.

8. State bans electronic balloting in 4 counties / Touch-screen firm accused of 'reprehensible,' illegal conduct [Електронний ресурс]. – Режим доступу: https://www.sfgate.com/politics/article/State-bans-electronic-balloting-in-4- counties-2784975.php.

9. Voting Machine Company Submits to Inquiry [Електронний ресурс]. – Режим доступу: https://www.nytimes.com/2006/10/31/us/politics/31vote.html?_r=1&oref=sl ogin.

10. Why machines are bad at counting votes [Електронний ресурс]. – Режим доступу: https://www.theguardian.com/technology/2009/apr/30/e-votingelectronic-polling-systems.

11. Baudron, O. Practical multi-candidate election system. In proceedings of the twentieth annual ACM symposium on Principles of distributed computing, // Baudron, O., Fouque, P.A., Pointcheval, D., Stern, J., Poupard, G. — ACM, 2001. -pp. 274–283.

12. Benaloh, J. Receipt-free secret-ballot elections. In Proceedings of the twenty-sixth annual ACM symposium on Theory of computing, // Benaloh, J., and Tuinstra, D. — ACM, - 1994. - pp. 544–553.

13. Card, D. Does voting technology affect election outcomes? touch-screen voting and the 2004 presidential election // Card, D., Moretti, E.- he Review of Economics and Statistics.,-2007.-pp. 660-673.

14. Card, D. Does voting technology affect election outcomes? touch-screen voting and the 2004 presidential election // Card, D., Moretti, E.- he Review of Economics and Statistics.,-2007.-pp. 660-673.

15. Cohen, J. D. A robust and verifiable cryptographically secure election scheme // Cohen, J. D., Fischer, M. J.- Yale University. Department of Computer Science, -1985.

16. Czepluch, J. S. The use of block chain technology in different application domains, // Czepluch, J. S., Lollike, N. Z., Malone, S. O. — The IT University of Copenhagen, Copenhagen, - 2015.

17. DeMillo, R. A. Proceedings of the fourteenth annual ACM symposium on Theory of computing // DeMillo, R. A., Lynch, N. A. - ACM,-1982.- pp. 383–400.

18. Fujioka, A. A practical secret voting scheme for large scale elections. In International Workshop on the Theory and Application of Cryptographic Techniques, // Fujioka, A., Okamoto, T., and Ohta, K. — Springer, - 1992. – pp. 244-251.

19. Hirt, M. Efficient receipt-free voting based on homomorphic encryption. In Advances in CryptologyEUROCRYPT 2000, // Hirt, M., and Sako, K. — Springer, - 2000. - pp. 539–556.

20. Jason, P. C. E-voting system based on the bitcoin protocol and blind signatures, // Jason, P. C., Yuichi, K.— TOM 10, - 2017.- pp. 14-22.

21. Jonker, H. Privacy and verifiability in voting systems: Methods, developments and trends, // Jonker, H., Mauw, S., and Pang, J. — Computer Science Review 10, - 2013. - pp. 1-30.

22. M. D. Pierro: What Is the Blockchain? [Text] / M. D. Pierro // Computing in Science & Engineering. – 2017. – PP. 92-95.

23. Peters, G. W. Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In Banking Beyond Banks and Money, // Peters, G. W., Panayi, E.— Springer, -2016. - pp. 239–278.

24. Sako, K. Receipt-free mix-type voting scheme. In Advances in CryptologyEUROCRYPT95, // Sako, K., and Kilian, J. — Springer, - 1995.- pp. 393–403.

25. John P Mechalas Introducing the intel® software guard extensions tutorial series [Електронний ресурс]. – Режим доступу: https://software.intel.com/content/www/us/en/develop/articles/introducing-the-intel-software-guard-extensions-tutorial-series.html.

26. Radhesh Krishnan Konoth, Emanuele Vineti MineSweeper: An In-depth Look into Drive-by Cryptocurrency Mining and Its Defense [Електронний ресурс]. – Режим доступу: https://dl.acm.org/doi/pdf/10.1145/3243734.3243858.

27. Abdul Wahab extending Hyperledger Fabric network: Adding a New Peer [Електронний ресурс]. – Режим доступу: https://wahabjawed.medium.com/extendinghyperledger-fabric-network-adding-a-new-peer-4f52f70a7217.

28. Hyperledger Fabric Documentation [Електронний ресурс]. – Режим доступу: https://hyperledger-fabric.readthedocs.io/ru/latest/whatis.html.

29. Victor Costan, Srinivas Devadas Intel SGX Explained [Електронний ресурс]. – Режим доступу: http://css.csail.mit.edu/6.858/2020/readings/costan-sgx.pdf.

30. Robert Watson, Trusted Code Base in a UNIX Environment [Електронний ресурс]. – Режим доступу: https://lists.freebsd.org/pipermail/trustedbsddiscuss/2000-April/000050.html.

31. Daniel Ehnas The Magic of Intel’s SGX [Електронний ресурс]. – Режим доступу: https://medium.com/magicofc/the-magic-of-intels-sgx-how-to-hello-it-sec-worldfb0295d6c33b.

32. Fan Zhang, Ethan Cecchetti Town Crier: An Authenticated Data Feed for Smart Contracts [Електронний ресурс]. – Режим доступу: https://towncrier.org/files/2016/168.pdf.

33. Thomas Knauth, Michael Steiner Integrating Intel SGX Remote Attestation with Transport Layer Security [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1801.05863.pdf.

34. Ferdinand Brasser, Urs Muller Software Grand Exposure: SGX Cache Attacks Are Practical [Електронний ресурс]. – Режим доступу: https://www.usenix.org/system/files/conference/woot17/woot17-paper-brasser.pdf.

35. Hyperledger Avalon Architecture Overview Revision 0.3 [Електронний ресурс]. – Режим доступу: https://hyperledger.github.io/avalon/docs/avalon-arch.pdf.

36. Rui Yuan, Hai-Bo Chen ShadowEth: Private Smart Contract on Public Blockchain [Електронний ресурс]. – Режим доступу: https://www.trustkernel.com/uploads/ pubs/shadoweth.pdf.

37. Dominik Meißner, Felix Engelmann PeQES: A Platform for Privacy-enhanced Quantitative Empirical Studies [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/2103.05544.pdf.

38. “Blockchain: The blockchain for beginners guide to blockchain technology and leveraging blockchain programming” Josh Thompsons 2017, - 84c.

39. “A Gentle Introduction to Blockchain” [Електронний ресурс]. – Режим доступу: https://medium.com/@mattmcilwham.

40. Smart Contracts: The Blockchain Technology That Will Replace Lawyers: [Електронний ресурс]. – Режим доступу: https://blockgeeks.com/guides/smart-contracts.

41. Blockchain Security Solutions [Електронний ресурс]. – Режим доступу: https://safenet.gemalto.com/blockchain.

42. Swanson T. Great Chain of Numbers: A Guide to Smart Contracts, Smart Property, and Trustless Asset Management. 2014.

43. Gisolfi D., Patel M., Radulovich R. Decentralized Identity Introduction. 2018 [Електронний ресурс]. – Режим доступу: https://www.ibm.com/downloads/cas/OPEQYEL7).

44. Kozin I.S. Providing personal data protection based on the block chain technology // Fourth Conference on Software Engineering and Information Management (SEIM-2019) P. 10–16.

45. Індекс NPS та навіщо його вимірювати [Електронний ресурс]. – Режим доступу: https://blog.integrica.com.ua/2021/11/30/indeks-nps-2.

46. Блокчейн і fintech: як змінюється сфера фінансів [Електронний ресурс]. – Режим доступу: https://www.epravda.com.ua/columns/2021/04/14/672973.

ДОДАТКИ

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

ЗАТВЕРДЖУЮ

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

_____ Юрій ЯРЕМЧУК
"24" вересня 2022 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:
«Підвищення достовірності системи конфіденційного голосування на основі смарт-контрактів в мережі блокчейн»

08-72.МКР.007.00.097.ТЗ

Керівник магістерської кваліфікаційної роботи
к.т.н., доцент
_____ Картінець В.В.

Вінниця – 2022 р.

1. Найменування та область застосування

Система голосування в мережі блокчейн із підвищеною достовірністю.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ №203 від 14. 09. 2022 р.

3. Мета та призначення розробки

3.1 Мета розробки: підвищення достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів.

3.2 Призначення: розроблений програмний продукт може використовуватися компаніями, які піклуються про безпеку транзакцій своїх клієнтів.

4. Джерела розробки

4.1. Kozin I.S. Providing personal data protection based on the block chain technology // Fourth Conference on Software Engineering and Information Management (SEIM-2019).

4.2. Smart Contracts: The Blockchain Technology That Will Replace Lawyers: [Електронний ресурс]. – Режим доступу: https://blockgeeks.com/guides/smart-contracts.

4.3. Документация 6и6лиотеки Web3py [Електронний ресурс] // Piper Merriam, Jason Carver. – Режим доступу: https://web3py.readthedocs.io/en/stable/

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків;

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок.

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

5.2.3 Програмний засіб повинен виконувати свої функції.

6. Вимоги до програмної документації

6.1 Обов'язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3

7. Вимоги до технічного захисту інформації

7.1 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

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

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

9. Стадії та етапи розробки

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Початок	Закінчення
1	Визначення напрямку магістерської роботи, формулювання теми	15.09.2022	30.09.2022
2	Аналіз предметної області обраної теми	01.10.2022	10.10.2022
3	Апробація отриманих результатів	11.10.2022	15.10.2022
4	Розробка алгоритму роботи	16.10.2022	31.10.2022
5	Написання магістерської роботи на основі розробленої теми	01.11.2022	15.11.2022
6	Розробка економічної частини	15.11.2022	23.11.2022
7	Передзахист магістерської кваліфікаційної роботи	24.11.2022	25.11.2022
8	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	26.11.2022	18.12.2022
9	Захист магістерської кваліфікаційної роботи	19.12.2022	21.12.2022

10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;

– програмний додаток;

– презентація;

– відзив керівника роботи;

– відзив опонента

Технічне завдання до виконання прийняв _____Манелюк В.Д.

Додаток Б. Лістинг програмного коду смарт-контракту

```
pragma solidity ^0.4.24;
contract Voting {
struct Start {
string title;
 string question;
string option1;
string option2;
string option3;
uint count1;
uint count2;
uint count3;
mapping (address => bool) voted;
bool exists;
}
address creator;
mapping (address => bool) public whitelist;
mapping (string => Start) polls;
function createVoting (string _question, string _name, string _option1, string _option2, string _option3) public {
polls[_name].title = _name;
polls[_name].question = _question;
polls[_name].option1 = _option1;
polls[_name].option2 = _option2;
polls[_name].option3 = _option3;
polls[_name].count1 = 0; polls[_name].count2 = 0;
 polls[_name].count3 = 0; polls[_name].exists = true;
creator = msg.sender;
}
function addWhitelistAddress (address _address) public {
if (msg.sender != creator) return;
whitelist[_address] = true;
}
function addManyWhitelistAddress (address[] _addresses) public {
if (msg.sender != creator) return;
for (uint256 i = 0; i < _addresses.length; i++) {
whitelist[_addresses[i]] = true;
}
}
```

```
function vote (string optionName, string pollName) public {
require(doesPollExist(pollName));
require(whitelist[msg.sender] == true);
require(hasAlreadyVoted(pollName));
polls[pollName].voted[msg.sender] = true;
if (keccak256(polls[pollName].option1) == keccak256(optionName)) {
polls[pollName].count1 += 1; }
if (keccak256(polls[pollName].option2) == keccak256(optionName)) { polls[pollName].count2 += 1;
}
if (keccak256(polls[pollName].option3) == keccak256(optionName)) {
polls[pollName].count3 += 1; } } function getPollName (string pollName) public view returns (string) {
require(doesPollExist(pollName));
return polls[pollName].question; }
function getOption1 (string pollName) public view returns (string) {
require(doesPollExist(pollName));
return polls[pollName].option1;
}
function getOption2 (string pollName) public view returns (string) {
require(doesPollExist(pollName));
return polls[pollName].option2; }
 function getOption3 (string pollName) public view returns (string) {
require(doesPollExist(pollName));
return polls[pollName].option3; }
function getPollCounts (string pollName) public view returns (uint[3]) {
require(doesPollExist(pollName));
return [polls[pollName].count1, polls[pollName].count2, polls[pollName].count3]; }
function doesPollExist (string pollName) private view returns (bool) {
if (polls[pollName].exists) { return true; }
else {
return false; }
}
function hasAlreadyVoted (string pollName) private view returns (bool) {
if (polls[pollName].voted[msg.sender]) { return false;
}
else { return true;
}}
}
```

Додаток В. Лістинг програмного коду захищеного середовища

```
package main
/* Imports
* 4 utility libraries for handling bytes, reading and writing JSON, formatting, and string manipulation
* 2 specific Hyperledger Fabric specific libraries for Smart Contracts
*/
import (
"bytes"
"encoding/json"
"fmt"
"strconv"
fc "github.com/chaincode/fabcrypt"
"github.com/hyperledger/fabric/core/chaincode/shim"
sc "github.com/hyperledger/fabric/protos/peer"
)
// Define the Smart Contract structure
 type SmartContract struct {
}

/* Define Records structure, with 4 properties.
Structure tags are used by encoding/json library
*/
type Records struct {
Fullname string `json:"fullname"`
DOB string `json:"dob"`
Insurance string `json:"insurance"`
Vaccine string `json:"vaccine"`
Status string `json:"status"`
}
/*
* The Init method *
called when the Smart Contract is instantiated by the network
* Best practice is to have any Ledger initialization in separate function
-- see initLedger()
*/
func (s *SmartContract) Init(APIstub shim.ChaincodeStubInterface) sc.Response {
        return shim.Success(nil)
}
/*
```

```
* The Invoke method *
called when an application requests to run the Smart Contract
The app also specifies the specific smart contract function to call with args
*/
func (s *SmartContract) Invoke(APIstub shim.ChaincodeStubInterface) sc.Response {

// Retrieve the requested Smart Contract function and arguments function, args := APIstub.GetFunctionAndParameters()
// Route to the appropriate handler function to interact with the ledger
if function == "queryPD" {
    return s.queryPD(APIstub, args)
} else if function == "initLedger" {
    return s.initLedger(APIstub)
} else if function == "recordPD" {
    return s.recordPD(APIstub, args)
} else if function == "queryAllPD" {
    return s.queryAllPD(APIstub)
} else if function == "changePD" {
    return s.changePD(APIstub, args)
}

return shim.Error("Invalid Smart Contract function name.")
}

 /*
* The queryPD method *
Used to view the records of one particular tuna
It takes one argument -- the key for the tuna in question
*/
func (s *SmartContract) queryPD(APIstub shim.ChaincodeStubInterface, args []string) sc.Response {
    if len(args) != 1 {
       return shim.Error("Incorrect number of arguments. Expecting 1")
}
     tunaAsBytes, _ := fc.Decrypter(APIstub, args[0])
     if tunaAsBytes == nil {
         return shim.Error("Could not locate tuna")
}
return shim.Success(tunaAsBytes)
}
```

```
/*
* The initLedger method *
Will add test data (10 records)to our network
*/
func (s *SmartContract) initLedger(APIstub shim.ChaincodeStubInterface) sc.Response {
    tuna := []Records{
        Records{Fullname: "Dylan Freedman", DOB:
"08.03.1996", Insurance: "Aflac:AOP1504XZ05A225", Vaccine: "Sputnik V", Status:"Well"},
        Records{Fullname: "Omar Garden", DOB: "02.04.1999", Insurance: "Aflac:R1C50405782DGL5",
Vaccine: "Sputnik V", Status:"Sick"},
        Records{Fullname: "Ivan Smith", DOB: "23.03.1976", Insurance: "Assurant:P1S4935TT170S567",
Vaccine: "Pfizer", Status:"Well"},
        Records{Fullname: "Anna Proper", DOB: "15.10.1983", Insurance: "Reso:TXX1496REWW105425",
Vaccine: "Moderna", Status:"Well"},
        Records{Fullname: "Alex Grey", DOB: "30.01.1985", Insurance: "Unum:1IDP49351SS230Y1", Vaccine:
"Pfizer", Status:"Sick"},
        Records{Fullname: "Tomas Edison", DOB: "02.01.2000", Insurance: "Unum:1W4E94G11BNM7101Q",
Vaccine: "AstraZeneca", Status:"Well"},
        Records{Fullname: "Mia Heigl", DOB: "24.12.1971", Insurance: "GEICO:VF49655F00104F301", Vaccine:
"Pfizer", Status:"Sick"},
        Records{Fullname: "Akay Fox", DOB: "19.07.1994", Insurance: "Omega:1445NHD8506D6L691A",
Vaccine: "Sputnik V", Status:"Well"},
        Records{Fullname: "Elvis Hughes", DOB: "28.09.1964", Insurance: "Reso:2F14S85153CVB091P",
Vaccine: "Sputnik V", Status:"Well"},
        Records{Fullname: "Tanya Makarova", DOB: "14.02.1973", Insurance: "Reso:PL1487A745DC091BN",
Vaccine: "Moderna", Status:"Well"},
    }
     i := 0
    for i < len(tuna) {
        fmt.Println("i is ", i)
    tunaAsBytes, _ := json.Marshal(tuna[i])
    fc.Encrypter(APIstub, strconv.Itoa(i+1), []byte(tunaAsBytes))
    fmt.Println("Added", tuna[i])
    i = i + 1
    }
    return shim.Success(nil)
    }
    /*
    * The recordPD method *
```

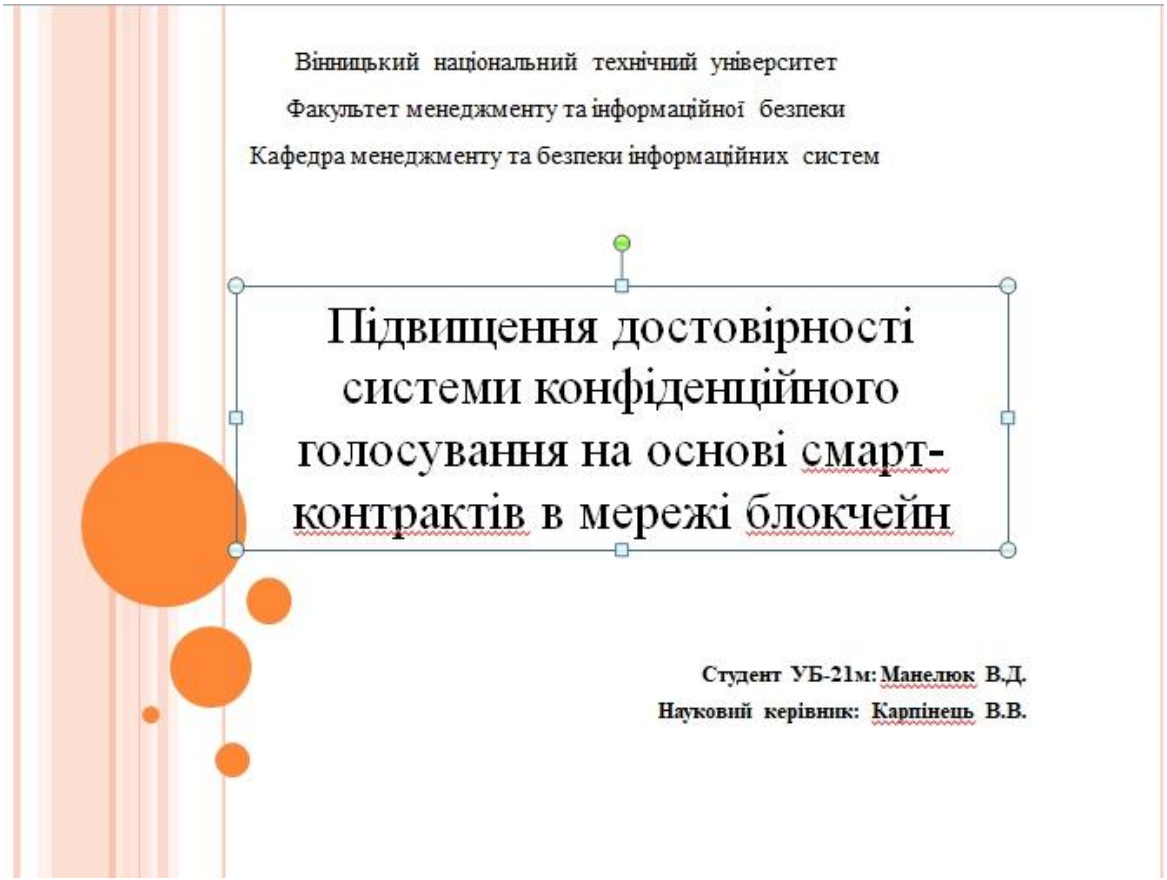
```
This method takes in five arguments (attributes to be saved in the ledger).
*/
func (s *SmartContract) recordPD(APIstub shim.ChaincodeStubInterface, args []string) sc.Response {
    if len(args) != 6 {
        return shim.Error("Incorrect number of arguments. Expecting 5")
}

var tuna = Records{ Fullname: args[1], DOB: args[2], Insurance: args[3], Vaccine: args[4], Status: args[5] }

tunaAsBytes, _ := json.Marshal(tuna)
error := fc.Encrypter(APIstub, args[0], []byte(tunaAsBytes))
if error != nil {
       return shim.Error(fmt.Sprintf("Failed to Encrypt tuna catch: %s", args[0]))
}
        return shim.Success(nil)
}
/*
* The queryAllPD method *
allows for assessing all the records added to the ledger(all tuna catches)
This method does not take any arguments. Returns JSON string containing results.
*/
func (s *SmartContract) queryAllPD(APIstub shim.ChaincodeStubInterface) sc.Response {
startKey := "0"
endKey := "999"
resultsIterator, err := APIstub.GetStateByRange(startKey, endKey)
if err != nil {
   return shim.Error(err.Error())
}
defer resultsIterator.Close()
// buffer is a JSON array containing QueryResults
var buffer bytes.Buffer
buffer.WriteString("[")

bArrayMemberAlreadyWritten := false
for resultsIterator.HasNext() {
    queryResponse, err := resultsIterator.Next()
    if err != nil {
        return shim.Error(err.Error())
}
```

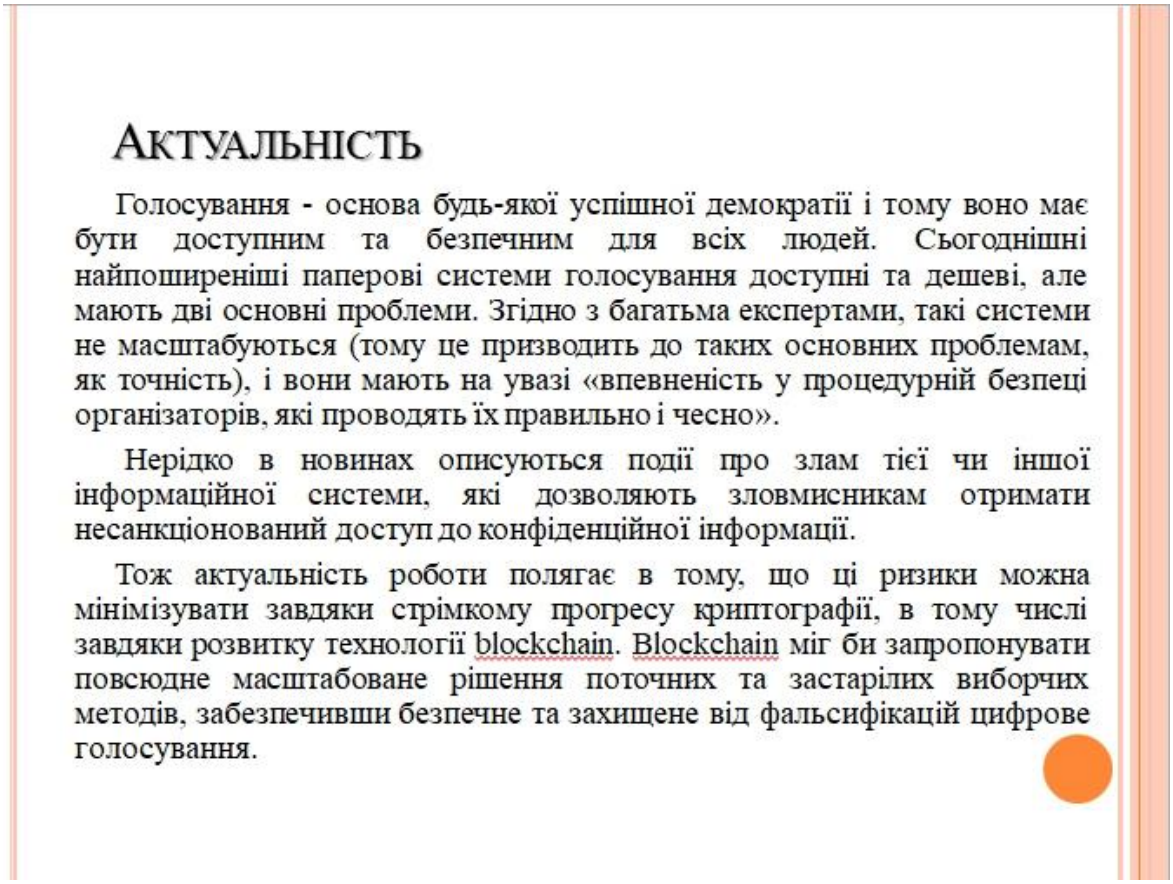
```
 // Add comma before array members,suppress it for the first array member
if bArrayMemberAlreadyWritten == true {
    buffer.WriteString(",")
}
buffer.WriteString("{\"Key\":")
buffer.WriteString("\"")
buffer.WriteString(queryResponse.Key)
buffer.WriteString("\"")

buffer.WriteString(", \"Record\":")
// Record is a JSON object, so we write as-is
buffer.WriteString(string(queryResponse.Value))
buffer.WriteString("}") bArrayMemberAlreadyWritten = true
}
buffer.WriteString("]")
fmt.Printf("- queryAllPD:\n%s\n", buffer.String())
return shim.Success(buffer.Bytes())
}
 /*
* The changePD method *
The data in the world state can be updated with who has possession.
This function takes in 2 arguments, tuna id and new holder name.
*/
func (s *SmartContract) changePD(APIstub shim.ChaincodeStubInterface, args []string) sc.Response {
    if len(args) != 2 {
        return shim.Error("Incorrect number of arguments. Expecting 2")
}
    tunaDec, error := fc.Decrypter(APIstub, args[0] )
    if error != nil {
        return shim.Error("Could not get or Decrypt tuna")
}

tuna := Records{}

json.Unmarshal(tunaDec, &tuna)
// Normally check that the specified argument is a valid holder of tuna
// we are skipping this check for this example
tuna.Fullname = args[1]
```

```
tunaDec, _ = json.Marshal(tuna)
    err := fc.Encrypter(APIstub, args[0], []byte(tunaDec))
if err != nil {
    return shim.Error(fmt.Sprintf("Failed to Encrypt changed tuna holder: %s", args[0]))
}
return shim.Success(nil)
}
/*
* Check hash of the Enclave SGX
*/
func (s *SmartContract) checkHash(APIstub shim.ChaincodeStubInterface, args []string) sc.Response
{
     if len(args) != 1 {
          return shim.Error("Incorrect number of arguments. Expecting 1")
}
tunaAsBytes, _ := fc.Decrypter(APIstub, args[0])
if tunaAsBytes == nil {
          return shim.Error("Could not locate tuna")
}
if(tunaAsBytes == args[1])
          return shim.Success(tunaAsBytes)
else return shim.Error("Invalid hash, enclave is not valid")
}
/*
    * main function *
calls the Start function
The main function starts the chaincode in the container during instantiation.
*/
func main() {
    // Create a new Smart Contract
    err := shim.Start(new(SmartContract))
     if err != nil {
          fmt.Printf("Error creating new Smart Contract: %s", err)
}}
```

```
echo "Removing key from key store..."
rm -rf ./hfc-key-store
cd chaincode
rm -rf errors
git clone https://github.com/pkg/errors.git
cd ..
cd basic-network
./start.sh

sudo docker ps -a

sudo docker-compose -f docker-compose.yml up -d cli

sudo docker ps -a

echo 'Installing chaincode..'
sudo docker exec -it cli peer chaincode install -n mycc -v 1.0 -p
"github.com/chaincode/"

echo 'Instantiating chaincode..'
sudo docker exec -it cli peer chaincode instantiate -o orderer.example.com:7050 -C mychannel -n mycc -v 1.0 -c
'{"Args":[]}'

echo 'Getting things ready for Chaincode Invocation..should take only 10 seconds..'

sleep 10
echo 'Initializing Ledger'

sudo docker exec -it cli peer chaincode invoke -o orderer.example.com:7050 -C mychannel -n mycc -c
'{"function":"initLedger","Args":[]}'
sleep 3
echo 'querying test record..'
sudo docker exec -it cli peer chaincode query -C mychannel -n mycc -c '{"function":"queryPD","Args":["3"]}'
echo 'querying all records..'
sudo docker exec -it cli peer chaincode query -C mychannel -n mycc -c '{"function":"queryAllPD","Args":[]}'
sleep 1
echo 'Success!'
exit 1
```

Додаток Г. Ілюстрований матеріал (презентація)





Мета:

Метою цієї дипломної роботи є підвищення достовірності системи електронного голосування в мережі блокчейн на основі смарт-контрактів.

Об’єкт та предмет

- **Об'єктом дослідження** цієї випускної кваліфікаційної роботи є технологія блокчейн, а також механізми передачі та обробки персональних даних, які забезпечують конфіденційність цих даних.
- **Предметом дослідження** є ступінь безпеки знайдених рішень, тобто наскільки запропоновані механізми запобігають крадіжці та розкриттю персональних даних.

Результат роботи

Результатом даної роботи стало створення покращеної системи голосування за рахунок її децентралізації та підключеного захищеного середовища, цим самим довівши її працездатність а також відкритість та прозорість, а, відповдно, і безпечність.

Основні задачі роботи:

- побудувати смарт-контракт на блокчейн мережі Ethereum;
- підключити до блокчейн мережі систему голосування з можливістю обробки даних у довіреному середовищі виконання (Trusted Execution Environment);
- налаштувати зв'язок між мережею та системою голосування, забезпечивши конфіденційність даних криптографічними методами;
- розробити графічний інтерфейс для взаємодії із системою голосування.

Вимоги до розробленої системи:

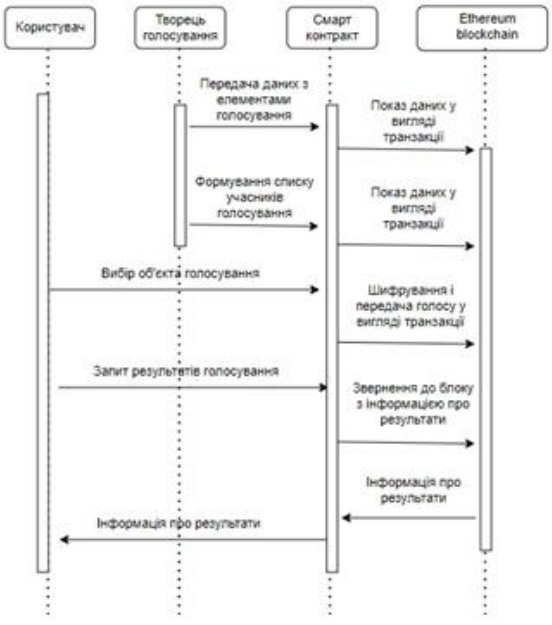
- можливість створити опитування та списки об'єктів голосування до них;
- реєстрація учасників для кожного опитування;
- можливість голосування;
- забезпечення прозорості;
- забезпечення відмовостійкості;
- відсутність можливості вносити несанкціоновані зміни, що впливають на результат голосування

Мови програмування

- Смарт-контракт, який реалізує основні процеси голосування, буде написаний на мові програмування Solidity
- інструмент Ganache-CLI, який дозволяє запустити у себе на пристрої симулятор блокчейна з увімкненим RPC протоколом
- Для взаємодії із децентралізованим вузлом криптовалюти Ethereum буде використовуватися інтерфейс для мови програмування Python – web3.py

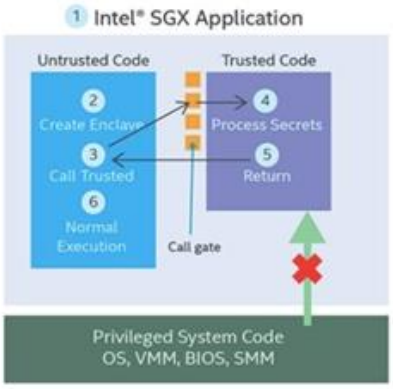
Модель розробленого додатку

Діаграма послідовності роботи системи електронного голосування



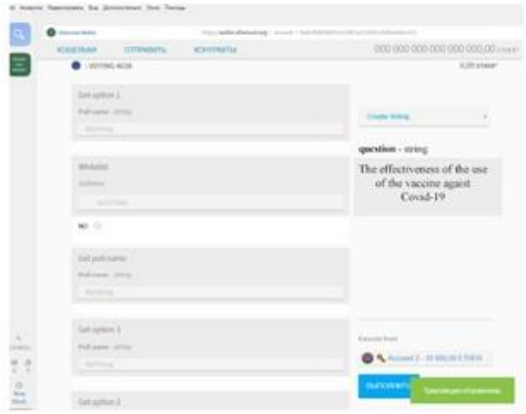
Діаграма варіантів використання системи голосування користувачем

Підключення через смарт-контракт виконання програми Intel SGX

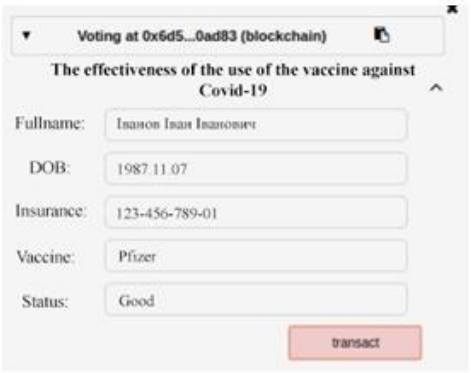


1. Поділ програми на два компоненти: довірений та ненадійний.
2. Створення анклаву у захищеному компоненті.
3. Передача виконання коду анклаву за допомогою довіреного виклику функції з ненадійної частини програми.
4. Код анклаву обробляє дані у відкритому вигляді, при цьому доступ до них із привілейованого рівня ОЗ заборонено.
5. Передача результатів роботи коду анклаву в ненадійну частину програми.
6. Виконання у звичайному режимі у ненадійному компоненті.

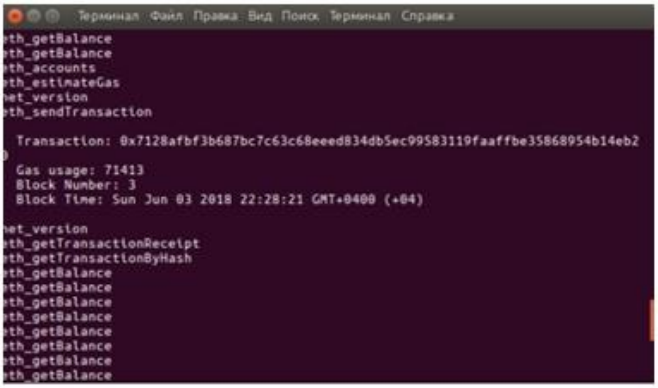
Взаємодія зі смарт-контрактом через браузер Mist



Заповнення даних для опитування



Транзакція про успішне надсилання інформації в блокчейн



Висновки

Метою даної випускної кваліфікаційної роботи було підвищення достовірності системи електронного голосування в мережі блокчейн із захистом їх конфіденційності на основі смарт-контрактів. У розробленій системі реалізовано функції голосування, реєстрація в системі реалізована шляхом створення облікового запису в мережі blockchain, в якій розгорнуть смарт-контракт та додаванням користувача у білий список того чи іншого створеного опитування.

Розроблена система голосування працює в blockchain мережі Ethereum та використовує створення анклавів Intel SGX для ізоляції персональних даних від ненадійної частини ОС. Використання надійних алгоритмів шифрування та схеми підпису забезпечує конфіденційність і цілісність критично важливих даних.

Економічний аналіз показників свідчить про доцільність впровадження системи голосуваня на основі смарт-контрактів в практиці.

Отже, завдання, поставлені на початку роботи були виконані, мета випускної кваліфікаційної роботи була досягнуто.

Дякую за увагу!

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ
ЗАПОЗИЧЕНЬ

Назва роботи: Підвищення достовірності системи конфіденційного голосування на основі смарт-контрактів у мережі блокчейн

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 98% Схожість 2 %

Аналіз звіту подібності (відмітити потрібне):

1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ (підпис) Коваль Н.П. (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ (підпис) Манелюк В.Д. (прізвище, ініціали)

Керівник роботи _____ (підпис) Карпінець В.В. (прізвище, ініціали)