Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Дослідження вразливостей та безпеки технології Blockchain

Виконав: студент IV курсу, групи СБс-41
спеціальності 125 Кібербезпека
(шифр і назва спеціальності)

(підпис) Цимбрак.І.С.
(прізвище та ініціали)

Керівник (підпис) Скарга-Бандурова І.С.
(прізвище та ініціали)

Нормоконтроль (підпис) Лобур Т.Б.
(прізвище та ініціали)

Завідувач кафедри (підпис) Загородна Н.В.
(прізвище та ініціали)

Рецензент (підпис)
(прізвище та ініціали)

Тернопіль - 2023

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)
за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)
Студенту Цимбраку Івану Сергійовичу
звище, ім'я, по батькові)
1. Тема роботи Дослідження вразливостей та безпеки технології Blockchain

Керівник роботи Скарга-Бандурова Інна Сергіївна, д.т.н., проф.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом ректора від « 04 » 04 2023 року № 4/7-349
2. Термін подання студентом завершеної роботи 21.06.2023 р.
3. Вихідні дані до роботи наукові літературні джерела

4. Зміст роботи (перелік питань, які потрібно розробити):
1. Аналіз предметної області.
2. Теоретична частина.
3. Практична частина.
4. Безпека життєдіяльності, основи хорони праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титулка. 2. Актуальність. 3. Мета, задачі дослідження. 4. Класична схема стеганографічного каналу 5. Порівняння стенографічних методів. 6. Моделі ШНМ. 7 Одна із можлививх варіацій ШНМ. 8. Загальна технологія роботи стегосистеми 9. Аналіз нейромережевих архітектур для генерації стеготекстів. 10, 11. Приклад апробації технології роботи стегосистеми 12. Основні результати проведеного дослідження

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці			

7. Дата видачі завдання 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	04.04 – 06.04	Виконано
2.	Підбір джерел про вразливості на основі блокчейн		
	технологій	07.04 – 11.04	Виконано
3.	Опрацювання джерел про вразливості технології		
	блокчейн	12.04 – 16.04	Виконано
4.	Виконання дослідження про вразливості технології		
	блокчейн	17.04 – 23.04	Виконано
5	Розроблення програмного коду для захисту	24.04 – 29.04	
6.	Оформлення розділу «Аналіз предметної області»	30.04 – 07.05	Виконано
7.	Оформлення розділу «Технологічні та організаційні		
	методи підвищення безпеки блокчейн-мережі»	08.05 – 15.05	Виконано
8.	Оформлення розділу «Аналіз алгоритмів шифрування		
	для підвищення безпеки блокчейн-мережі»	16.05 – 21.05	Виконано
9.	Виконання завдання до підрозділу «Безпека		
	життєдіяльності, основи охорони праці»	14.05 – 21.05	Виконано
10.	Оформлення кваліфікаційної роботи	22.05 – 05.06	Виконано
11.	Нормоконтроль		
12.	Перевірка на плагіат		
13.	Попередній захист кваліфікаційної роботи		
14.	Захист кваліфікаційної роботи		

Студент _____ Цимбрак.І.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Скарга-Бандурова І.С.

АНОТАЦІЯ

Дослідження вразливостей та безпеки технології Blockchain // Кваліфікаційна робота ОР «Бакалавр» // Цимбрак Іван Сергійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра кібербезпеки, група СБс–41 // Тернопіль, 2023 // С. – 68, рис. – 6, таб. – 1, слайдів – 12, бібліогр. – 12, додат. - 1.

Ключові слова: АЛГОРИТМ, АТАКА, БЛОКЧЕЙН, БЕЗПЕКА, ВРАЗЛИВІСТЬ, ХЕШ- ФУНКЦІЯ.

Кваліфікаційна робота "Дослідження вразливостей та безпеки технології Blockchain" є всебічним аналізом потенційних слабких місць та стратегій забезпечення безпеки в системах, що базуються на технології блокчейн, а також можливостей використання блокчейну у різних галузях.

Робота охоплює широкий спектр тем, починаючи з аналізу загальних концепцій та структур блокчейн, до специфічних вразливостей, таких як атака 51%, що має особливе значення для мереж, основаних на доказу роботи (Proof of Work). Робота детально досліджує цю та інші потенційні атаки, а також надає реалізацію алгоритму PoS для їх запобігання.

В роботі представлено інтеграцію технологічних та організаційних методів підвищення безпеки блокчейн-мереж з алгоритмами шифрування найвищої криптографічної стійкості.

Ця робота представляє важливий внесок у галузі дослідження безпеки блокчейну, ідентифікуючи потенційні вразливості та надаючи конструктивні рекомендації щодо їх вирішення. Вона є цінним джерелом для дослідників, розробників та інших професіоналів, що працюють у цій області, а також для всіх, хто цікавиться блокчейн технологією та її використанням.

ABSTRACT

Research on vulnerabilities and security of Blockchain technology // Bachelor's Degree Qualification Work // Ivan Serhiyovych Tsimbrak // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cyber Security group SBs–41 // Ternopil, 2023 // P. – 68, fig. – 6, tab. – 1, slides – 12, biblio. – 12, app. - 1.

Keywords: ALGORITHM, ATTACK, BLOCKCHAIN, SECURITY, VULNERABILITY, HASH FUNCTION.

The bachelor's degree work "Research on vulnerabilities and security of Blockchain technology" provides a comprehensive analysis of potential weak spots and security strategies in systems based on blockchain technology, as well as the possibilities of using blockchain in various fields.

The paper covers a wide range of topics, from an analysis of general concepts and structures of the blockchain to specific vulnerabilities, such as a 51% attack, which is particularly important for networks based on proof of work (PoW). The paper thoroughly investigates this and other potential attacks and provides an implementation of the PoS algorithm to prevent them.

The paper presents the integration of technological and organizational methods of enhancing the security of blockchain networks with encryption algorithms of the highest cryptographic resilience.

This work represents a significant contribution to the field of blockchain security research, identifying potential vulnerabilities and providing constructive recommendations for their resolution. It serves as a valuable resource for researchers, developers, and other professionals working in this field, as well as for anyone interested in blockchain technology and its application.

ЗМІСТ

ВСТУП 7
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 9
1.1 Основні поняття технології блокчейну 9
1.2 Принципи роботи блокчейну 11
1.3 Протоколи консенсусу в блокчейні 14
1.4 Технологічні та організаційні методи підвищення безпеки блокчейн-мереж 15
1.4.1 Шифрування 17
1.4.2 Підписи 17
1.4.3 Організаційні методи підвищення безпеки 18
2ТЕХНОЛОГІЧНІ ТА ОРГАНІЗАЦІЙНІ МЕТОДИ ПІДВИЩЕННЯ БЕЗПЕКИ БЛОКЧЕЙН-МЕРЕЖ 20
2.1 Підвищення криптостійкості за допомогою хеш-функції 20
2.2 Алгоритм MD5 23
2.3 Алгоритм SHA-1 26
2.4 Аналіз методів підвищення криптографічної стійкості 29
ЗАНАЛІЗ АЛГОРИТМІВ ШИФРУВАННЯ ДЛЯ ПІДВИЩЕННЯ БЕЗПЕКИ БЛОКЧЕЙН-МЕРЕЖ 32
3.1 Основні види вразливостей для блокчейн-мереж 32
3.2 Аналіз типових атак на блокчейн 34
3.2.1 Атака 51% 35
3.2.2 Атака силового повторення (Replay Attack): 37
3.2.3 Атака Sybil 38
3.2.4 Атака Eclipse 39
3.2.5 Атака на протоколи консенсусу 41
3.3 Аналіз вразливостей смарт-контрактів 43
3.3.1 Зловживання цільовим кодом 44
3.3.2 Помилки в коді 45

3.3.3 Відмова в роботі 46
3.3.4 Оракул-атаки 48
3.4 Реалізація алгоритму PoS для запобігання атаки 51% 49
4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ 52
4.1 Долікарська допомога при харчових отруєннях 52
4.1.1 Ознаки та симптоми харчових отруєнь 52
4.1.2 Принципи надання першої долікарської допомоги при харчових отруєннях 53
4.1.3 Медична допомога при харчових отруєннях 54
4.1.4 Профілактика харчових отруєнь 55
4.1.5 Як блокчейн може допомогти в сфері харчових отруєнь 56
4.2 Заходи щодо автоматизації виробничих процесів, які сприяють покращенню умов праці 57
4.2.1 Автоматизація важких та небезпечних задач 58
4.2.2 Покращення точності та консистентності 59
4.2.3 Використання блокчейн в заходах щодо автоматизації виробничих процесів, які сприяють покращенню умов праці 60
ВИСНОВКИ 61
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 63
ДОДАТОК А. Код запобігання атаки 51% 66

ВСТУП

Актуальність роботи обумовлена тим, що у сучасному світі технології блокчейн набувають все більшої популярності та широкого застосування у різних галузях. Відмінність блокчейну від традиційних технологій полягає в його децентралізованому характері, що дозволяє забезпечити надійність, прозорість та безпеку обміну даними. Кібербезпека - найперспективніша галузь використання цієї технології. За допомогою блокчейна можна створити унікальний ключ для автентифікації, який підтверджуватиметься всіма користувачами мережі. Підміна швидко розкриється, що дозволить системі миттєво відреагувати на спробу вторгнення. Це стосується як блокчейн-гаманців, так і інших систем — банківських, освітніх, державних та корпоративних. За оцінками фахівців, масове впровадження технології зменшить ймовірність злому на 70–85%. Оскільки в мережі кожен учасник зберігає копію масиву даних, видалити інформацію повністю майже неможливо. Завдяки цьому скорочується час відновлення системи після атаки та зменшується ймовірність отримання незворотної шкоди. Однак, незважаючи на численні переваги, технологія блокчейн також має свої вразливості, які можуть бути використані зловмисниками для проведення атак та компрометації мережі.

Мета роботи - аналіз технології блокчейн, її вразливостей та безпеки, а також розробка рекомендацій щодо підвищення рівня захисту блокчейн-мереж. У роботі розглядаються основні принципи та типи блокчейн-мереж, протоколи консенсусу, а також види вразливостей та загрози, що можуть виникнути під час роботи з блокчейном.

Щоб досягти встановленої цілі, необхідно виконати наступні завдання:

– необхідно дослідити принципи функціонування технології блокчейн і визначити різновиди вразливостей, що можуть виникнути;

– проаналізувати технологічні та організаційні методи підвищення безпеки блокчейн-мереж;

– дослідити алгоритми шифрування;

– реалізувати метод підвищення криптографічної стійкості.

Об'єкт роботи – безпека блокчейн-мережи.

Основним об'єктом дослідження є методи та алгоритми шифрування, які сприяють підвищенню безпеки блокчейн-мереж.

У цьому дослідженні була використана методологія, яка базується на методах теорії ігор та криптографії, для проведення необхідних досліджень.

Наукова новизна отриманих результатів досліджень у інтеграції технологічних та організаційних методів підвищення безпеки блокчейн-мереж з алгоритмами шифрування найвищої криптографічної стійкості.

Цінність отриманих результатів полягає у їх унікальності та значущості для науки комплексний підхід обумовлює стабільність та забезпечення цілісності даних блокчейн-мережі в умовах постійно змінного кіберпростору.

1 АНАЛІЗ ТЕХНОЛОГІЇ BLOCKCHAIN

1.1 Основні поняття технології блокчейну

Блокчейн – це децентралізована, розподілена база даних, що функціонує за принципом зв'язного списку блоків, які містять інформацію про транзакції. Кожен блок містить унікальний хеш-код, що посилається на попередній блок, тим самим створюючи ланцюг. Це структура дозволяє забезпечити високий рівень безпеки, оскільки змінити інформацію в одному з блоків без зміни всіх наступних блоків практично неможливо.

Блокчейн, або блоковий ланцюг, є видом розподіленої бази даних, в основі якої лежать криптографічні принципи. Основними компонентами блокчейна є блоки, які містять групу транзакцій, і ланцюги, що з'єднують ці блоки за допомогою криптографічних хешів. Основні принципи роботи блокчейна включають наступні пункти:

- Децентралізація: у традиційних базах даних одна централізована організація контролює всі дані. В той же час, блокчейн є розподіленою базою даних, яка не має центрального контролю. Замість цього, всі учасники мережі мають копії всього блокчейна, що гарантує прозорість та відмовостійкість.
- Транзакції і блоки: кожна транзакція, що відбувається в мережі, зазначається в блоку. Коли блок заповнюється транзакціями, він додається до ланцюга блоків. Цей процес відбувається за допомогою механізму, відомого як майнінг.
- Майнінг: майнінг - це процес валідації та додавання нових блоків до блокчейна. Майнери конкурують між собою, виконуючи складні обчислення для знаходження наступного блоку. Переможець отримує винагороду в формі криптовалюти.
- Консенсус: для додавання блоку до ланцюга необхідно досягнути консенсусу між учасниками мережі. Різні блокчейни використовують різні

алгоритми консенсусу, такі як Proof-of-Work (PoW), Proof-of-Stake (PoS), Delegated Proof-of-Stake (DPoS), та інші.

- Незмінність: однією з ключових характеристик блокчейна є його незмінність. Однією з важливих переваг цього аспекту є те, що однією з принципових переваг цього аспекту є те, що транзакції, що вже були записані в блокчейн, не можна змінити або видалити.
- Анонімність та прозорість: блокчейн забезпечує анонімність для своїх користувачів, оскільки транзакції здійснюються за допомогою криптографічно зашифрованих адрес, а не імен користувачів. Однак, в той же час, всі транзакції є публічно відкритими, що забезпечує високий рівень прозорості.

Ці принципи визначають роботу блокчейну і створюють умови для його використання в різних сферах, починаючи від фінансових послуг і закінчуючи логістикою і управлінням постачаннями.

Блокчейн-мережі можна класифікувати за різними критеріями, але найоширенішим поділом є на публічні, приватні та консорціуми (або змішані) мережі.

- Публічні блокчейни: публічні блокчейни, як правило, відкриті для всіх користувачів, які хочуть приєднатися. У них немає централізованого керівництва, і всі учасники мають однакові права. Будь-хто може брати участь в процесі майнінгу, проводити транзакції та перевіряти їх. Bitcoin та Ethereum - два найвідоміших приклади публічних блокчейнів.
- Приватні блокчейни: приватні блокчейни, відомі також як дозволені мережі, обмежені дозволами та контролюються одним або кількома організаціями. Учасники мережі вибираються адміністратором, який також встановлює правила для учасників. Ці мережі часто використовуються внутрішньо в організаціях для покращення ефективності та прозорості.
- Консорціуми (змішані) блокчейни: консорціуми представляють собою золоту середину між публічними та приватними блокчейнами. Вони

керуються групою організацій, які визначають правила мережі. Консорціуми зазвичай використовуються у спільних проектах між різними організаціями.

1.2 Принципи роботи блокчейну

Блокчейн одна із видів ширшого класу технологій зберігання та синхронізації даних – розподіленого реєстру (англ. DLT – Distributed Ledger Technology). Ключова властивість всього класу технологій розподіленого реєстру – відсутність централізованого управління. Кожен елемент мережі розподіленої системи, який складається з програмного забезпечення та реєстру, самостійно виконує записи у своєму реєстрі незалежно від інших вузлів і підтримує синхронізацію з ними у рамках однорангової мережі. Одним з основних аспектів блокчейну як виду розподіленого реєстру є з'єднання записів в послідовний ланцюжок блоків з використанням криптографічних алгоритмів, що призводить до його назви - "блокчейн" (англ. blockchain, ланцюжок блоків).

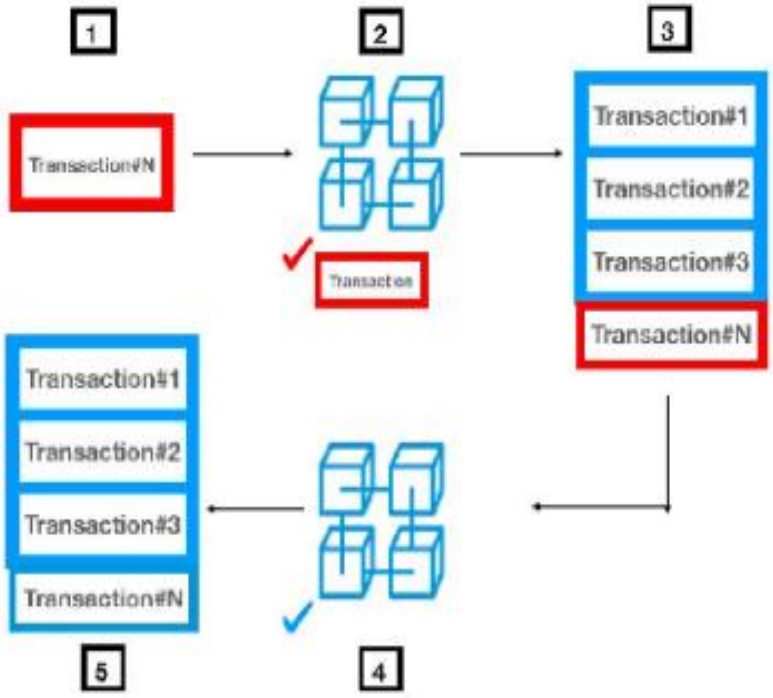

Рисунок 1.1 – Принцип роботи реєстру блокчейну

На рис.1.1 представлені етапи роботи реєстру блокчейну:

- Підготовка транзакції: на цьому етапі сторона А створює транзакцію, яка включає інформацію, включаючи публічну адресу одержувача, цифровий підпис джерела і повідомлення про транзакцію. Тепер ця транзакція стала доступною для всіх вузлів у блокчейні.
- Перевірка транзакції: вузли блокчейну працюють у моделі без довіри, де кожен вузол (машина, на якій працює клієнтське програмне забезпечення блокчейну) отримує цю транзакцію та перевіряє цифровий підпис за допомогою відкритого ключа сторони А. Після успішної перевірки ця автентифікована транзакція зберігається в черзі реєстру та чекає, поки всі вузли успішно перевірять ту саму транзакцію.
- Генерація блоків: транзакції в черзі впорядковуються разом, і один із вузлів мережі створює блок. У блокчейні біткойн/біткойни отримують винагороду, коли біткойн-вузол, також відомий як майнер, створює блок, вирішуючи якусь математично складну задачу.
- Перевірка блоку: після успішного створення блоку вузли в мережі обробляються для ітеративного процесу перевірки, у якому більшість вузлів мають досягти консенсусу.
- З'єднання блоків: після успішного механізму консенсусу блоки перевіряються та додаються до блокчейну.

Блокчейн є децентралізованою базою даних, де всі записи збираються в блоки та посилаються один на одного за допомогою криптографічних методів. Кожен блок містить записи (транзакції), ідентифікатор блоку та хеш-суми попередніх блоків. Ці хеш-суми визначаються за допомогою криптографічних хеш-функцій. Завдяки використанню хеш-функцій у поєднанні з розподіленою архітектурою, блокчейн забезпечує незмінність та незворотність всього ланцюжка блоків та транзакцій.

У блокчейні також велику роль відіграє алгоритм консенсусу - набір математичних правил і функцій. Основним завданням алгоритму консенсусу є

генерація та синхронізація ланцюжка блоків між усіма учасниками мережі. База даних блокчейну зберігається в необмеженій кількості учасників блокчейн-мережі, які можуть бути спеціальними вузлами консенсусу або майнінговими вузлами. Учасники мережі в блокчейні є невідомими заздалегідь та можуть приєднуватись або відключатись у будь-який момент. Алгоритм консенсусу забезпечує досягнення загальної згоди серед усіх учасників мережі щодо стану бази даних. Вузли консенсусу також відповідають за групування нових транзакцій в блоки та обчислення хеш-функцій для захисту ланцюжка блоків. Як правило, ці вузли отримують винагороду у формі цифрових активів (наприклад, криптовалюти) за свою роботу. Всі ці особливості, включаючи економічні мотиваційні механізми для учасників блокчейн-мереж, сприяли створенню спеціалізованих алгоритмів консенсусу, відмінних від консенсусів розподілених систем, що використовувалися до появи блокчейну.

Об'єднання властивостей розподіленого реєстру разом з блоковою структурою даних, яка ґрунтується на криптографічній зв'язаності, дозволяє блокчейну успішно здійснювати два з трьох ключових аспектів інформаційної безпеки - цілісність та доступність інформації.. У силу децентралізованої топології та криптографічних механізмів, зловмисні маніпуляції інформацією стають вкрай дорогими та скрутними, а сама інформація залишається доступною для всіх учасників за значних змін у розмірах блокчейн-мережі. Однак, традиційна модель децентралізованої публічної блокчейн-мережі, яка забезпечує прозорість та стійкість до цензури, не забезпечує третій аспект інформаційної безпеки - конфіденційність даних, через свою архітектуру та ідеологію. Це привело до появи приватного блокчейну як моделі, яка спрямована на вирішення проблем масштабованості та забезпечення конфіденційності даних.

Приватний блокчейн принципово відрізняється від публічного блокчейну за своєю моделлю доступу до мережі, де право внесення змін до реєстру належить обмеженій групі учасників. Також приватний блокчейн

відрізняється від громадського блокчейну якісно. У приватному блокчейні з'являється оператор, що робить мережу розподіленою, а не децентралізованою. Однак, приватний блокчейн дозволяє забезпечити конфіденційність записів, оскільки доступ до них контролюється політиками безпеки. Ці приватні мережі стають все більш популярними як інфраструктура для корпоративних та державних завдань.

Існує модель гібридного блокчейна, що поєднує обидва підходи. При ній записи з приватної мережі або їх метадані можуть зберігатися в публічному блокчейні, забезпечуючи додаткову відмовостійкість всього реєстру.

З точки зору безпеки блокчейн варто оцінювати не як самостійну технологію, а як інфраструктурний шар для конкретного сценарію – базу даних для корпоративної інформаційної системи, виконання децентралізованого додатку або смарт-контракту тощо. Аналіз численних інцидентів інформаційної безпеки, пов'язаних з блокчейн-рішеннями, показує, що часто найуразливішою частиною є не блокчейн-мережа, а суміжні компоненти та інформаційні системи.

1.3 Протоколи консенсусу в блокчейні

Протоколи консенсусу відіграють важливу роль у блокчейн-технології, оскільки вони дозволяють забезпечити децентралізацію, надійність та безпеку мережі. Ось деякі з основних протоколів консенсусу, які використовуються в різних блокчейн-платформах:

- Proof-of-Work (PoW): цей протокол консенсусу вимагає від майнерів виконувати складні математичні завдання для створення нових блоків та додавання їх до ланцюга. Першим, хто вирішить завдання, виплачується винагорода у вигляді криптовалюти. Bitcoin та Ethereum (наразі) використовують PoW.
- Proof-of-Stake (PoS): у протоколі PoS, учасники мережі валідують транзакції на основі кількості власних монет, які вони забезпечують (ставки).

Чим більше монет має валідатор, тим більше ймовірно, що він буде обраний для створення нового блоку. PoS вважається екологічнішим та енергоефективнішим, порівняно з PoW. Ethereum планує перехід на PoS через оновлення Ethereum 2.0.

- Delegated Proof-of-Stake (DPoS): DPoS - це варіант PoS, в якому валідатори обираються учасниками мережі на основі кількості делегованих монет. Обрані валідатори отримують право створювати нові блоки та підтверджувати транзакції. DPoS використовується в деяких блокчейн-проектах, таких як EOS та Lisk.
- Practical Byzantine Fault Tolerance (PBFT): PBFT - це протокол консенсусу, який працює на основі повідомлень між вузлами мережі для досягнення згоди. Він забезпечує високу ефективність та безпеку, навіть у разі наявності ворожих або нечесних вузлів. PBFT використовується в деяких блокчейн-проектах, таких як Hyperledger Fabric та Stellar.

Окрім цих основних протоколів консенсусу, існує багато інших алгоритмів, які розробляються для покращення швидкості, масштабованості та безпеки блокчейн-мереж. Вибір протоколу консенсусу залежить від потреб конкретної блокчейн-платформи та її використання.

Наведенні протоколи консенсусу використовуються для забезпечення безпеки, стабільності та децентралізації в блокчейн-мережах. Вони допомагають учасникам домовитися про правильність транзакцій та блоків, а також забезпечують, що ніхто не зможе змінити дані без згоди інших учасників мережі.

1.4 Технологічні та організаційні методи підвищення безпеки блокчейн-мереж

1.4.1 Шифрування

Шифрування є одним з ключових технологічних методів підвищення безпеки блокчейн-мережі. Воно забезпечує конфіденційність, цілісність та автентичність даних, переданих та збережених в мережі. Шифрування застосовується для захисту приватних ключів користувачів, транзакцій та інших важливих даних.

Основні аспекти шифрування в блокчейн-мережі:

- Асиметричне шифрування: використання пари ключів (приватного та публічного) для захисту даних. Відомо також як криптосистема з відкритим ключем. Асиметричне шифрування використовується для генерування цифрових підписів та забезпечення автентичності транзакцій.
- Симетричне шифрування: використання одного секретного ключа для шифрування та дешифрування даних. Це шифрування зазвичай швидше за асиметричне, але може мати проблеми з розподілом ключів.
- Хеш-функції: криптографічні хеш-функції використовуються для створення унікального, фіксованого розміру виводу (хеш) з вхідних даних. Вони важливі для підтвердження цілісності даних та створення відповідної структури блокчейн.
- Стійкість до квантових атак: оскільки розвиток квантового обчислення може призвести до зламу традиційних криптографічних алгоритмів, необхідно розглядати можливість впровадження стійких до квантових атак шифрувальних схем. Квантостійке шифрування включає алгоритми, які можуть протистояти злому за допомогою квантових комп'ютерів, забезпечуючи безпеку даних в майбутньому.
- Гомоморфне шифрування: це вид шифрування, який дозволяє виконувати обчислення безпосередньо на зашифрованих даних без необхідності їх дешифрування. Гомоморфне шифрування може бути корисним для підвищення приватності та захисту даних у блокчейн-мережах, де потрібно обробляти конфіденційну інформацію.

- Застосування шифрування на рівні платформи: Для забезпечення загального захисту даних, шифрування може бути впроваджено на різних рівнях архітектури блокчейн-мережі, таких як шифрування на рівні баз даних, шифрування мережевих з'єднань або шифрування застосунків.

Застосування сучасних та надійних методів шифрування є важливою складовою підвищення безпеки блокчейн-мережі. Від розробників та адміністраторів мережі вимагається постійний моніторинг нових криптографічних розробок, щоб забезпечити своєчасне оновлення алгоритмів та підтримку безпеки на відповідному рівні.

1.4.2 Підписи

Криптографічні підписи є суттєвим елементом безпеки блокчейн-мереж. Вони використовуються для забезпечення аутентифікації, цілісності та невідкликаності транзакцій та інших даних, які передаються в мережі.

Основним типом криптографічних підписів, що використовується в блокчейн-технологіях, є цифровий підпис. Цифрові підписи засновані на асиметричному криптосистемі (публічному та приватному ключам), яка гарантує, що лише власник приватного ключа може підписати транзакцію, а будь-хто з публічним ключем може перевірити підпис.

Ефективність цифрових підписів залежить від сили криптографічного алгоритму, який використовується для генерації ключів та підписів. Деякі з найбільш поширених алгоритмів, що використовуються в блокчейн-мережах, включають ECDSA (еліптичні криві дійсної криптосистеми), EdDSA (Edwards-цільова криптосистема) та RSA (або Rivest-Shamir-Adleman криптосистема).

Щоб забезпечити більш високий рівень безпеки та відповідність сучасним стандартам, розробники блокчейн-мереж і смарт-контрактів повинні вибирати надійні алгоритми цифрових підписів та регулярно оновлювати свої криптографічні схеми з урахуванням нових загроз та відкриттів в області

криптографії.

1.4.3 Організаційні методи підвищення безпеки

Організаційні методи підвищення безпеки стосуються не тільки технічних рішень, але й процесів, політик та освіти, які можуть допомогти забезпечити безпеку блокчейн-мережі та її користувачів. Ось деякі з найбільш важливих організаційних методів для підвищення безпеки:

- Розробка та впровадження політик безпеки: забезпечити розробку та впровадження чітких політик безпеки, які враховують особливості блокчейн-мережі. Це може включати процедури аутентифікації користувачів, управління доступом, зберігання та передачі даних та реагування на інциденти безпеки.
- Регулярний аудит безпеки та оцінка ризиків: проводити регулярні аудити безпеки для виявлення потенційних слабких місць та ризиків у блокчейн-мережі. Включати аналіз коду, оцінку алгоритмів та інфраструктури, а також розглядати можливі людські ризики, такі як неправильне використання або втрата приватних ключів.
- Навчання та освіта користувачів: освіта та навчання користувачів щодо безпечного використання криптографічних технологій та протоколів мають велике значення для забезпечення стійкості та безпеки блокчейн-мережі. Навчальні програми та ресурси можуть допомогти користувачам зрозуміти основи криптографії, а також розвивати правильні звички щодо зберігання та використання приватних ключів.
- Безпека інфраструктури: забезпечити захист фізичної та віртуальної інфраструктури, пов'язаної з блокчейн-мережею. Це включає захист від несанкціонованого доступу, відмовостійкість, резервне копіювання даних та відновлення від аварійних ситуацій. Перевірити, що сервери, мережеві пристрої та інша інфраструктура знаходяться в безпечних місцях та налаштовані відповідно до найкращих практик безпеки.

- Розподіл відповідальності та ролей: встановити чіткі ролі та обов'язки для різних учасників блокчейн-мережі, що стосуються безпеки. Це може включати відповідальність за управління ключами, проведення аудитів безпеки, розробку політик безпеки та реагування на інциденти безпеки.
- Управління інцидентами та реагування на загрози: розробити процедури та плани реагування на різні типи інцидентів безпеки, такі як крадіжка ключів, атаки на інфраструктуру або витоки даних. Це включає плани на випадок відновлення після аварійних ситуацій, виявлення та сповіщення про загрози, а також координацію з іншими учасниками мережі для забезпечення ефективного реагування.
- Відкритість та співпраця зі спільнотою: співпраця з іншими учасниками блокчейн-спільноти, такими як розробники, експерти з безпеки, користувачі та регуляторні органи, може підвищити безпеку мережі. Участь у конференціях, семінарах та онлайн-форумах допоможе підтримувати відкритий діалог та взаємне навчання.
- Прозорість та відстежуваність: забезпечити прозорість операцій блокчейн-мережі та забезпечити відстежуваність даних та транзакцій. Це допоможе виявити аномалії та можливі випадки зловживань, а також підвищити довіру до мережі серед користувачів та регуляторів.
- Застосування регулятивних вимог та стандартів: дотримуватись відповідних законодавчих норм та міжнародних стандартів безпеки, щоб впевнитися, що блокчейн-мережа відповідає вимогам інформаційної безпеки та захисту даних. Регулярно перевіряти зміни у законодавстві та стандартах, щоб своєчасно вносити необхідні зміни у систему безпеки.

Дотримання організаційних методів підвищення безпеки допоможе забезпечити стійкість блокчейн-мережі від зовнішніх та внутрішніх загроз та забезпечить безпечне та надійне середовище для користувачів.

2 АНАЛІЗ АЛГОРИТМІВ ШИФРУВАННЯ ДЛЯ ПІДВИЩЕННЯ БЕЗПЕКИ БЛОКЧЕЙН-МЕРЕЖ

2.1 Підвищення криптостійкості за допомогою хеш-функції

В блокчейні використовується контрольна сума чи хеш. Це зашифрований заголовок, який надається масиву даних. Якщо змінити хоч один символ, хеш також зміниться - це стане помітним під час порівняння.

Блокчейн-платформа розрахована на постійне додавання нової інформації. Для цього і потрібний ланцюжок блоків. Оскільки інформація згодом додається, до одного масиву даних має прикріплюватись інший. Блок №2 містить не лише свою контрольну суму, а й хеш блоку №1. Відповідно, блок №3 включає хеш блоку №2 і так далі. Забравши одну ланку, розірветься весь ланцюг.

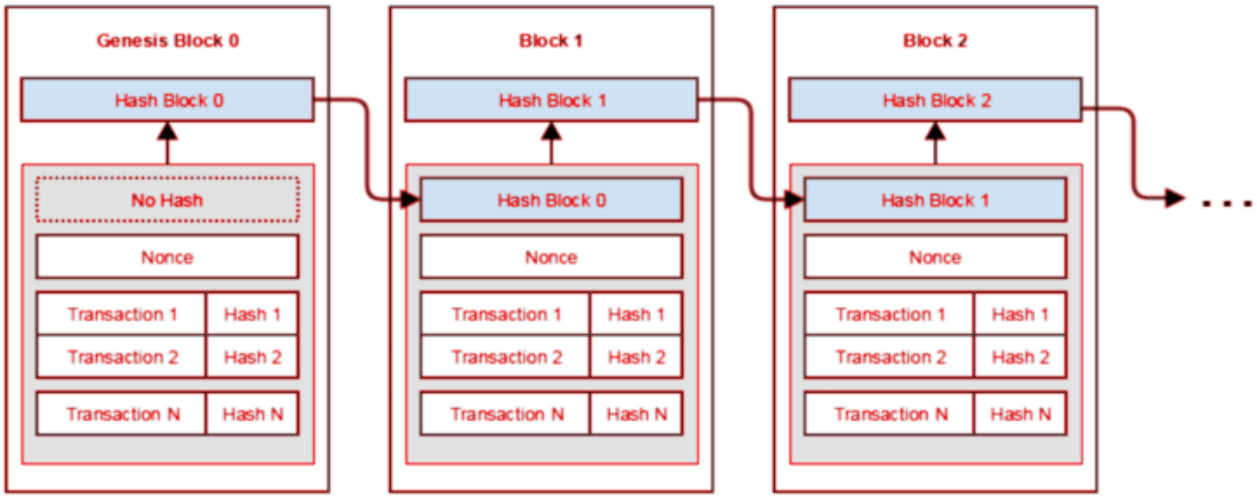

Рисунок 2.1 – Принцип формування хешу в блокчейні

Хеш-функція, також відома як функція згортки, є математичною операцією, яка перетворює масив вхідних даних будь-якої довжини у фіксований вихідний бітовий рядок за допомогою певного алгоритму.

Криптографічна хеш-функція – хеш-функція, що є криптографічно

стійкою, тобто відповідає ряду вимог, специфічних для криптографічних додатків.

У якості криптографічних генераторів псевдовипадкових чисел для створення декількох ключів на основі одного секретного ключа та для захисту інформації від несанкціонованого доступу використовуються криптографічно стійкі хеш-функції, такі як MD5, SHA1 та SHA-2.

Хеш-функція, яка має криптографічну стійкість, перетворює вихідний текст довільної довжини в один рядок фіксованої довжини. Цей рядок, який називається хешем, є унікальним для кожного вхідного тексту. Важливою властивістю хеш-функції є неможливість (або дуже висока складність) відновити вихідний текст з отриманого хешу. Ця хеш-функція-перетворення тексту визначається як:

$$H = hash(P), \tag{2.1}$$

де P - пароль (відкритий текст), довжина P від 0 до нескінченності; H-хеш (хешований текст), довжина $H = N$ біт (за умови що функція $hash$ повертає хеш-значення довжиною N біт).

Для посилення криптографічної стійкості використовують ітераційний алгоритм отримання ключів. Загальну схему ітераційного алгоритму можна представити як на рис. 2.2.

Ключі знаходяться за наступним алгоритмом:

$$\begin{gathered} K_1 = hash(PASSWORD) \\ K_2 = f(K_1) \\ \dots \\ K_n = f(K_{n-1}), \end{gathered} \tag{2.2}$$

де f-функція перетворення ключа.

Якщо знати K_1, то можливо обчислити всі інші K_i, $i = 2..n$.

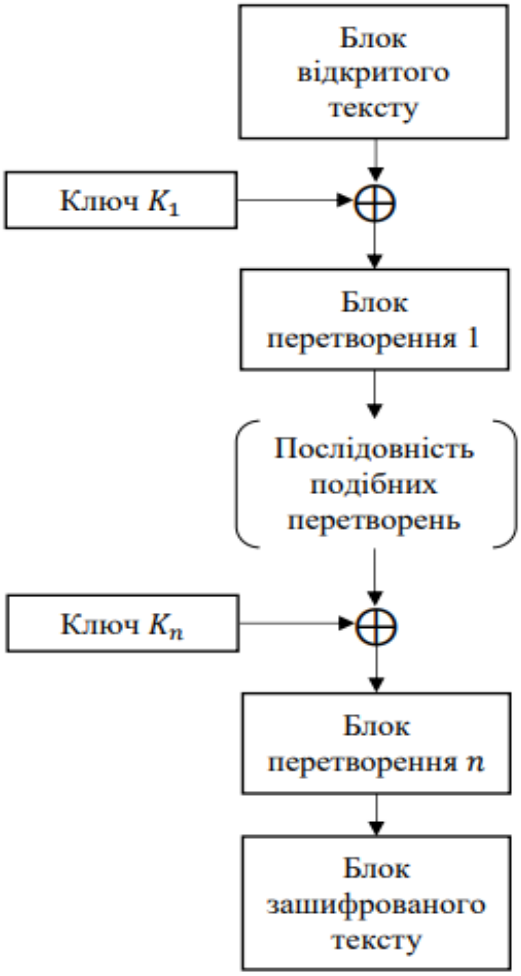

Рисунок 2.2 – Узагальнена схема ітераційних криптоалгоритмів

Зловмисник не знаючи первинного тексту (паролю), але якщо він має отриману конечну хеш-функцію, може розшифрувати і отримати вхідний текст. Для підбору пароля методом жорсткого паролю, потрібно зробити підбір 2^s значень (S-розмір хеш-значення в бітах). Якщо хеш-функція має розмір у 64 біт, тоді потрібно зробити перебір 2^{64} значень.

Приймаємо, що комп'ютер в середньому може перебирати 1000000 паролів в секунду. Тоді знаходження первинного тексту за його хеш-значення максимально займе 213 503 982 днів.

Якщо застосовувати альтернативний метод атаки, наприклад, пошук колізій хеш-функцій, кількість можливих варіантів буде зменшуватися в середньому вдвічі, тобто виходить $2^{64/2} = 2^{32}$ значень і тоді на пошук

відповідного хешу вийде всього лише приблизно 1,2 години.

2.2 Алгоритм MD5

Хеш-функція призначена для згортки вхідного масиву будь-якого розміру в бітову рядок, для MD5 довжина вихідний рядки дорівнює 128 бітам. Найчастіше хеш-функції використовуються для перевірки унікальності пароля, файлу, рядка і т.д. Використання MD5 дозволяє порівняти дайджест повідомлення з опублікованими, щоб переконатися, що дане повідомлення повністю збігається з оригінальним, тобто, не було пошкоджено або змінено. Дана процедура порівняння називається "перевірка хеша" (hashcheck).

Після отримання вхідного потоку даних, для якого потрібно знайти хеш, починається процес підготовки цього потоку до обчислення хешу. Довжина повідомлення позначається як L, де L - ціле невід'ємне число, яке може мати будь-яку довжину, включаючи нульову. Необов'язково, щоб довжина повідомлення була кратною будь-якому числу. Після надходження даних йде процес підготовки потоку до обчислень.

Алгоритм складається з п'яти кроків:

- Append Padding Bits

У вихідний рядок дописують одиничний байт 0x80, а потім дописують нульові біти, до тих пір, поки довжина повідомлення не буде порівнянна з 448 по модулю 512. Тобто дописуємо нулі до тих пір, поки довжина нового повідомлення не буде дорівнювати:

$$L = (512 * N + 448), \quad (2.3)$$

де L – довжина повідомлення, N - будь-яке натуральне число, таке, що цей вислів буде найближче до довжини блоку.

- Append Length

Далі в повідомлення дописується 64-бітове представлення довжини вихідного повідомлення.

- Initialize MD Buffer

На цьому кроці ініціалізується буфер.

$$word\ A\ 01\ 23\ 45\ 67$$

$$word\ B: 89\ ab\ cd\ ef$$

$$word\ C: fe\ dc\ ba\ 98$$

$$word\ D: 76\ 54\ 32\ 10$$

Як можна помітити буфер складається з чотирьох констант, призначений для збору хешу.

- Process Message in 16-Word Blocks

На четвертому кроці в першу чергу визначається 4 допоміжні логічні функції, які перетворять вхідні 32-бітові слова, як не дивно, в 32-бітові вихідні, які реалізують раунди шифрування.

I. $function\ F\ (X, Y, Z) = (X \cap Y) \cup (\overline{X} \cap Z)$

II. $function\ G\ (X, Y, Z) = (X \cap Z) \cup (\overline{Z} \cap Y)$

III. $function\ H\ (X, Y, Z) = X \oplus Y \oplus Z$

IV. $function\ I\ (X, Y, Z) = Y \oplus (\overline{Z} \cup X)$

Під час цього етапу також використовується "білий шум" - покращення алгоритму, яке включає масив з 64 елементів. Цей масив містить псевдовипадкові числа, які залежать від значення синуса числа i:

$$T[i] = int(2^{32} * |\sin(i)|). \quad (2.4)$$

Далі копіюємо кожен 16-бітний блок в масив $X[16]$ і виробляємо маніпуляції:

$$AA = A$$

$$BB = B$$

$$CC = C$$

$$DD = D$$

Вирівняні дані розбиваються на блоки (слова) по 32 біта, і кожен блок проходить 4 раунди з 16 операторів. Всі оператори однотипні і мають вигляд: $[abcd\ ksi]$, який визначається як:

$$A = B + ((A + function\ (B, C, D) + X\ [k] + T\ [i]) <<< s), \qquad (2.5)$$

A, B, C, D – регістри, $function\ (B, C, D)$ - одна з логічних функцій, $X[k]$ - k-тий елемент 16-бітного блоку, $T[i]$ - i-тий елемент таблиці «білого шуму», $<<< s$ - операція циклічного зсуву на s позицій вліво.

Тобто кожний раунд має вигляд:

Раунд 1

/*[abcd k s i] a = b + ((a+ F(b,cd) + X[k] + T[i]) <<< s). */
[ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]
[ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]
[ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]
[ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]

Раунд 2

/*[abcd k s i] a = b + ((a+ G(b,cd) + X[k] + T[i]) <<< s). */
[ABCD 1 4 17] [DABC 6 11 18] [CDAB 11 14 19] [BCDA 0 20 20]
[ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]
[ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]
[ABCD 14 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]

Раунд 3

/*[abcd k s i] a = b + ((a+ H(b,cd) + X[k] + T[i]) <<< s). */
[ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 34] [BCDA 14 23 36]
[ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]
[ABCD 13 4 41] [DABC 0 11 42] [CDAB 3 15 43] [BCDA 6 23 44]
[ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]

Раунд 4

/*[abcd k s i] a = b + ((a+ I(b,cd) + X[k] + T[i]) <<< s). */
[ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]
[ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]

[ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60] [ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]

Підсумовуємо результати обчислень:

$$A = A + AA$$

$$B = B + BB$$

$$C = C + CC$$

$$D = D + DD$$

- Output

Виводячи побайтово буфер ABCD, починаючи з A і закінчуючи D, отримаємо хеш.

Після завершення обробки поточного блоку необхідно перевірити, чи є ще блоки, які потребують обчислень. Якщо є ще блоки, то переходимо до наступного елементу масиву (i + 1) і повторюємо цикл обробки.

Цей алгоритм створює хеш і зміна хоча б одного елементу чи символу у первинному тексті кардинально змінює кінцевий результат. Але є один нюанс, це виникнення колізій.

Колізія хеш-функції – виникнення у пари незалежних об’єктів однакового хешу, тобто $H(x) = H(y)$. Колізії існують майже для будь-яких хеш-функцій, але залежно від якості хеш-функцій частота їх виникнення приблизно мінімальна.

2.3 Алгоритм SHA-1

Алгоритм SHA-1 (Secure Hash Algorithm Version 1) - безпечний алгоритм хешування, версія 1, був розроблений в далекому 1995 році. Для вхідного повідомлення довільної довжини (до 2 ексабайт) алгоритм генерує 160-бітове хеш-значення (дайджест повідомлення). Алгоритм отримує на вході повідомлення максимальної довжини 264 біт і створює в якості виходу

дайджест повідомлення довжиною 160 біт (рис. 2.4).

Алгоритм SHA-1 складається з таких етапів:

- Додавання відсутніх бітів і вказівка довжини

Текст, який має бути оброблений, розділяється на блоки по 512 біт. Якщо довжина тексту не є кратною 512 бітам, то він вирівнюється шляхом додавання спеціальних бітів.

Вирівнювання розпочинається додавання біта зі значенням 1 до кінця тексту. Після цього додається m нульових бітів, щоб довжина тексту стала кратною 512 бітам. Нарешті, до кінця тексту додається 64-бітне представлення довжини вихідного повідомлення.

- Ініціалізація буфера

В алгоритмі використовується 160-бітний буфер для зберігання проміжних і остаточних результатів хеш-функції. Ініціалізується п'ять 32-бітових робочих змінних A, B, C,D, E:

$$A = 67\ 45\ 23\ 01_{16};$$

$$B = EF\ CD\ AB\ 89_{16};$$

$$C = 98\ BA\ DC\ FE_{16};$$

$$D = 10\ 32\ 54\ 76_{16};$$

$$E = C3\ D2\ E1\ F0_{16}.$$

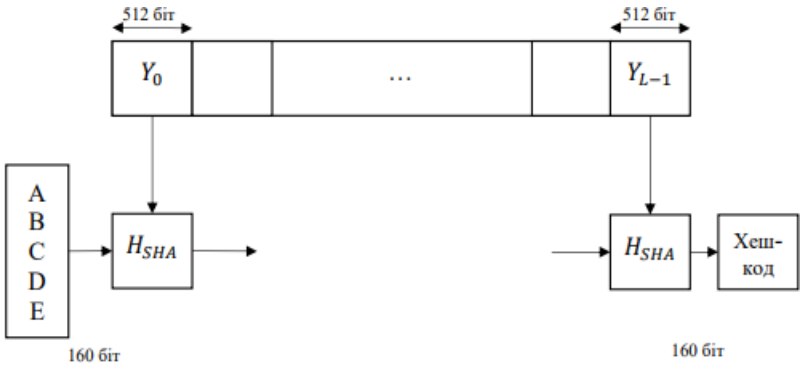

Рисунок 2.4 – Логіка виконання SHA-1

Вектор ініціалізації, що подається на вхід 1-го раунду результат конкатенації:

$$SHA0 = A \| B \| C \| D \| E \tag{2.6}$$

- Обробка повідомлення в 512-бітних блоках

Виконується обробка чергових 512 біт вихідного тексту. Для цього значення змінних A, B, C, D, E копіюються в змінні a, b, c, d, e і далі для t від 1 до 80 виконується перетворення значень даних змінних за схемою, зображеної на рисунку 2.5.

Кожна з 80 ітерацій може бути записана наступним чином:

$$TMP \leftarrow (a <<< 5) + f_t(b, c, d) + e + W_t + K_t;$$

$$e \leftarrow d;$$

$$d \leftarrow c;$$

$$c \leftarrow (b <<< 30);$$

$$b \leftarrow a;$$

$$a \leftarrow TMP;$$

де [+] - операція додавання за модулю 2^{32} , $f_t(X, Y, Z)$ - нелінійна функція, що має такий вигляд:

$$ft(X,Y,Z)\begin{cases}(X \cap Y) \cup (\underline{X} \cap Z), & 0 < t < 20; \\ X \oplus Y \oplus Z, (X \cap Y) \cup (X \cap Z) \cup (Y \cap Z), & 20 \le t < 40; \\ X \oplus Y \oplus Z, & 40 \le t < 59; \\ & 60 \le t \le 79; \end{cases} \tag{2.7}$$

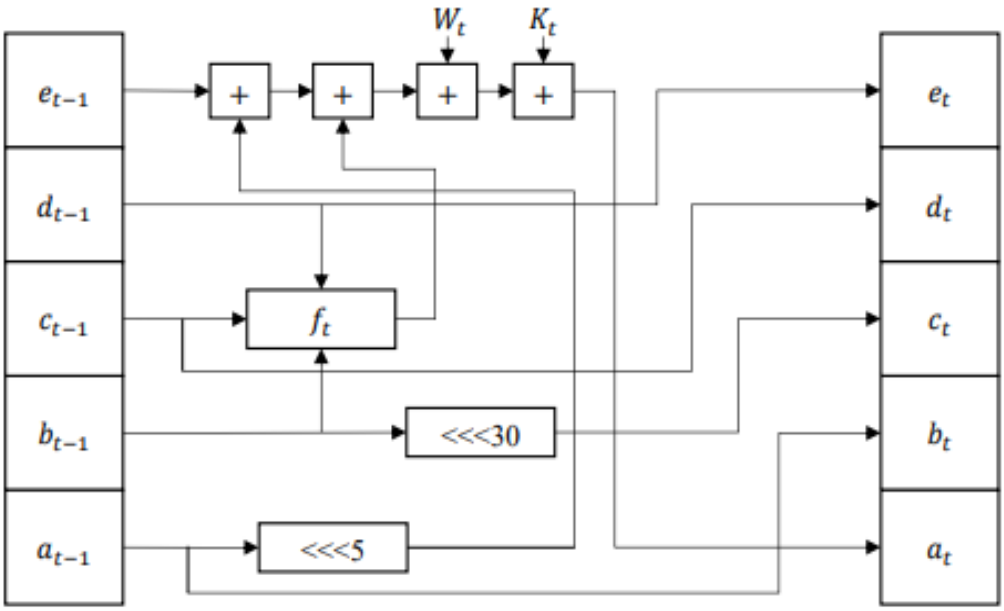

Рисунок 2.5 – Схема ітерації алгоритму SHA-1

Параметр K_t приймає чотири різних значення в залежності від номера поточної ітерації:

$K_t = 5A82799916,\ 0 < t < 19;$

$K_t = 6ED9EBA116,\quad 20 < t < 39;$

$K_t = 8F1BBCDC16,\quad 40 < t < 59;$

$K_t = CA62C1D616,\quad 60 < t < 79;$

де W_t - одне з шістнадцяти 32-бітних слів 512-бітного блоку повідомлення при $0 < t < 15$ або значення, яке визначається відповідно до наступного виразу при $15 < t < 80$:

$$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16} <<< 1 \qquad (2.8)$$

Значення змінних a, b, c, d, e незалежно один від одного складаються по модулю 2^{32} зі значеннями змінних A, B, C, D, E, в які потім і поміщаються отримані результати. Етап 3 виконується до тих пір, поки не буде оброблений весь текст. Після обробки останнього блоку тексту значення хеш-образу формується як $ABCDE$.

2.4 Аналіз методів підвищення криптографічної стійкості

Проведемо аналіз найбільш поширених хеш-функцій. Для пароля «Password» були отримані наступні хеш-значення (табл. 2.1).

Алгоритми CRC-32, Base-64, DES (Unix) використовувати не бажано, так як згенеровані хеш-функції мають замалий хеш, тобто дуже легко піддаються злому. Алгоритми MD2 і MD4 не є актуальними, як і попередні версії алгоритму SHA. Алгоритм Whirlpool попри те, що є надійним, є недоліки при обмеженої кількості «раундів» шифрування. З претендентів на реальне використання можна розглядати такі алгоритми серії SHA-512, MD5 та Whirpool.

Таблиця 2.1 - Отримання хешу для паролю «Password» за допомогою хеш-функцій

№ досліду	Хеш-функція	Хеш-значення
1	SHA-1	8be3c943b1609fffbfc51aad666d0a04adf83c9d
2	SHA-256	e7cf3ef4f17c3999a94f2c6f612e8a888e5b1026878e4e193 98b23bd38ec221a
3	SHA-3	690ace5d07a99566b98f337abba767367756f78bc0a867c9 24b1f99c4bc469473a0cf69f108b329b44887a32abc23254 d253d30db62538f0ebe72b0dabbed0c1
4	SHA-512	e6c83b282aeb2e022844595721cc00bbda47cb24537c177 9f9bb84f04039e1676e6ba8573e588da1052510e3aa0a32a 9e55879ae22b0c2d62136fc0a3e85f8bb
5	MD2	9dc7dd5f9b5f681a133e64c0089330e7
6	MD4	f15abd57801840f3348ddccafb677f6a
7	MD5	dc647eb65e6711e155375218212b3964
8	CRC-32	9abfd710
9	BASE-64	UGFzc3dvcmQ=
10	Tiger-128	78db95bcce175ab632095962774965d1
11	Whirlpool	fd07ba63996cdcfa6130ee82acec65da7487f51564bb7c6ea d6dabc6b9e8eac974e5d852edc545804ae68fa46fc59d478 9acab50bbc22b26fb24412f8dc11cde
12	DES(Unix)	Q37hnXsCsGjCU

На підставі проведеного аналізу реалізуємо метод підвищення криптографічної стійкості, де хеш-функцію можна використовувати для запобігання спаму електронною поштою. Алгоритм Hashcash, що лежить в

основі цієї ідеї, називається алгоритмом “Proof-of-Work”, і саме він утворює ядро Bitcoin. Загальна ідея полягає в тому, щоб приймати лише електронні листи, хеші яких задовольняють обмеження. Це змушує відправника електронного листа виконати певну обчислювальну роботу перед надсиланням електронного листа.

Наприклад, зміст листа матиме наступне хеш-значення:

```
# From: <Alice> alice@example.com
# Subject: Have you heard about Bitcoin?
# Hey Bob,I think you should learn about Blockchains! I've been investing in Bitcoin and currently have exactly 12.03 BTC in my account.

#hash:
71890dc61c21370874d2a7b74064396cb613a1924f09aa06925abc7842e6802c
```

Алгоритм Hashcash застосовує обмеження, що хеш-значення має бути нижчим за певне число:

```
from hashlib import sha256
secret_phrase = "bolognese"

def get_hash_with_secret_phrase(input_data, secret_phrase):
    combined = input_data + secret_phrase

    return sha256(combined.encode()).hexdigest()
```

Робота підтверджується відображенням певного хешу, і сервер може легко це перевірити:

```
email_body = "Hey Bob, I think you should learn about Blockchains! I've been investing in Bitcoin and currently have exactly 12.03 BTC in my  account."

print(get_hash_with_secret_phrase(email_body, secret_phrase))
```

3 АНАЛІЗ ВИДІВ ВРАЗЛИВОСТЕЙ ТА ЗАГРОЗ НА БЛОКЧЕЙН

3.1 Основні види вразливостей для блокчейн-мереж

Вразливості блокчейну - це уразливі місця в його дизайні та реалізації, які можуть бути використані зловмисниками для проведення атак. Вони можуть стосуватися як самого протоколу блокчейну, так і прикладних програм, які працюють на його основі, таких як смарт-контракти та криптовалютні гаманці.

Однією з ключових властивостей блокчейну є його децентралізованість, яка вимагає високого рівня взаємного довіря між учасниками мережі. Це може створювати певні вразливості, якщо учасники мережі використовують свої привілеї в мережі неналежним чином або якщо зловмисник здатний провести атаку, використовуючи вразливості в протоколі. Вразливість в контексті блокчейн-технологій відноситься до слабких місць, які можуть бути експлуатовані зловмисниками, щоб завдати шкоди мережі або її користувачам. Вразливості можуть виникати через помилки у коді, недоліки в протоколах або через неефективні механізми захисту. Загрози для блокчейн-мереж включають атаки, які мають на меті порушити безпеку, конфіденційність або цілісність мережі. Відмінність між вразливістю та загрозою полягає в тому, що вразливість є потенційним слабким місцем, яке може бути використано зловмисниками, тоді як загроза представляє реальну можливість використання цих вразливостей для завдання шкоди. Загрози можуть виникати внаслідок зовнішніх атак або внутрішніх факторів, таких як дії недобросовісних учасників мережі або відмова відповідного обладнання.

Деякі з найпоширеніших видів вразливостей та загроз для блокчейн-мереж включають:

– Масштабування: проблеми масштабування можуть виникнути через обмеження пропускної здатності мережі, обробки транзакцій та зберігання даних. Це може призвести до зниження продуктивності та підвищення витрат на обслуговування мережі.

– Приватність: недостатнє захищення приватності користувачів може призвести до витоку даних, а також зловмисники можуть аналізувати транзакції для виявлення відомостей про користувачів та їхню діяльність.

– Централізація: хоча блокчейн має децентралізовану природу, деякі аспекти, такі як майнінг, можуть призвести до концентрації влади у руках невеликої кількості учасників, що ставить під загрозу децентралізацію та безпеку мережі.

– Відмова в роботі: відмова в роботі може виникнути через атаки на вузли мережі, недоліки в алгоритмах консенсусу або відмову обладнання. Це може призвести до збою в роботі мережі або затримок в обробці транзакцій.

Розробка та застосування нових методів захисту, таких як криптографічні протоколи, розподілені архітектури, та використання новітніх алгоритмів консенсусу, можуть допомогти зменшити ризики вразливостей та загроз:

– Софт-форки та хард-форки: зміни у протоколах блокчейну можуть призвести до розділу мережі на дві або більше версій. Це може спричинити проблеми зі сумісністю, нестабільність та збої в роботі мережі.

– Зловживання смарт-контрактами: необхідно аналізувати вразливості смарт-контрактів та вдосконалювати їх безпеку, оскільки зловмисники можуть використовувати їх для маніпуляцій, крадіжок коштів або відмови в роботі.

– Зміни в регуляторному середовищі: зміни законодавства або регулятивних вимог можуть вплинути на роботу блокчейн-мережі та її учасників. Необхідно відстежувати та адаптуватися до регуляторних змін для забезпечення відповідності блокчейн-мережі до вимог законодавства.

– Компрометація приватних ключів: втрата або крадіжка приватних ключів може призвести до втрати контролю над активами користувача. Забезпечення безпеки приватних ключів та підвищення обізнаності користувачів про кращі практики зберігання ключів є важливими аспектами захисту блокчейн-мережі.

Інновації в галузі криптографії, розподіленого обчислення, приватності даних та інших сферах пов'язаних з блокчейн, можуть допомогти відкрити нові можливості та застосування для цієї технології. У свою чергу, це може привести до створення нових продуктів та послуг, які будуть відповідати потребам сучасного світу, забезпечуючи безпеку та прозорість для користувачів.

Упровадження більш стійких та надійних рішень щодо безпеки також зміцнить довіру спільноти до блокчейн-технологій, сприяючи їх широкому розповсюдженню в різних галузях економіки та суспільства. Постійна взаємодія між розробниками, експертами з кібербезпеки, учасниками ринку та регуляторами допоможе забезпечити стабільний розвиток та безпеку блокчейн-мереж, що в свою чергу може стати каталізатором для глобальної трансформації технологій та їх застосування в майбутньому.

Архітектура, криптографічні функції та алгоритми консенсусу децентралізованої блокчейн-мережі також можуть бути потенційно проексплуатовані, тим самим наражаючи на базовий принцип блокчейна – незмінність даних.

3.2. Аналіз типових атак на блокчейн

Різні види атак на блокчейн-мережі можуть впливати на їх надійність, прозорість та безпеку. Ось деякі з найпоширеніших атак та їх вплив на блокчейн-технології:

— Атака 51%: ця атака відбувається, коли один або група майнерів контролюють більше 51% хешрейту мережі. Це може дозволити їм маніпулювати блокчейном, включаючи двійні витрати.

— Атака силового повторення (Replay Attack): цей тип атаки відбувається, коли транзакція, яка була вже проведена в одній мережі, повторно відтворюється в іншій.

— Атака Sybil: ця атака полягає в створенні великої кількості псевдонімів або вузлів у мережі з метою вплинути на децентралізовані системи, які використовують систему голосування.

— Атака Eclipse: у цій атаці зловмисник намагається взяти під контроль всі з'єднання вузла, що дозволяє йому контролювати інформацію, яку цей вузол отримує.

— Атаки на смарт-контракти: смарт-контракти можуть містити вразливості у своєму коді, які можуть бути використані зловмисниками для крадіжки коштів або маніпуляції даними.

Розуміння цих вразливостей та загроз є важливим кроком у розробці заходів щодо забезпечення безпеки блокчейн-мереж.

3.2.1 Атака 51%

Атака 51% - це коли один або група майнерів мають контроль над більш як 51% загальної потужності обчислень в мережі блокчейну. З контролем над більшістю потужності мережі, ці майнери мають можливість виконувати маніпуляції з транзакціями: здійснювати двойний витрат (double spending), блокувати підтвердження нових транзакцій іншими учасниками мережі, або змінювати послідовність транзакцій.

Прикладом атаки 51% може бути ситуація, що сталася з криптовалютою Ethereum Classic у січні 2019 року. Невідомі хакери здійснили атаку 51% на мережу Ethereum Classic і зуміли подвійно витратити більш ніж $1.1 млн в ETC. В результаті атаки, криптобіржі, які обслуговували ETC, зазнали великих втрат. Така атака стала можливою, тому що Ethereum Classic має значно меншу хешрейт-потужність, ніж основна мережа Ethereum, і тому була більш вразливою до такого типу атак. Після атаки команда Ethereum Classic виконала ряд оновлень, спрямованих на підвищення безпеки мережі для запобігання подібних атак у майбутньому. Цей приклад демонструє, що, хоча атака 51% вимагає

значних ресурсів для її виконання, вона все ще можлива, особливо для блокчейн-мереж з невеликою хешрейт-потужністю. Тому важливо розробляти стратегії безпеки, які враховують цю можливість.

Вирішення проблеми атаки 51% є одним з найбільш обговорюваних питань в галузі блокчейн. Декілька потенційних рішень включають:

— Перехід на альтернативний протокол консенсусу: перехід від Proof of Work до альтернативних протоколів консенсусу, таких як Proof of Stake (PoS) або Delegated Proof of Stake (DPoS), може зменшити ймовірність атаки 51%, оскільки у них власникам великої кількості монет потрібно мати велику кількість монет, а не майнінгову потужність.

— Розподілене голосування: іншим можливим рішенням може бути впровадження системи розподіленого голосування, де вузли голосують за дійсність блоку. Це може допомогти забезпечити, що один учасник не може контролювати більшість мережі.

— Підвищення кількості учасників в мережі: чим більше учасників в мережі, тим складніше здійснити атаку 51%. Для цього потрібно стимулювати участь нових учасників і розширення мережі.

— Застосування технологій штучного інтелекту та машинного навчання: ці технології можуть використовуватися для виявлення аномалій та підозрілих активностей в мережі, що можуть вказувати на спробу атаки 51%.

Це лише декілька з представлених рішень, які можуть забезпечити вирішення проблеми атаки 51%. У будь-якому випадку, необхідна постійна відповідність і реагування на зміни в ландшафті блокчейн-технологій, щоб забезпечити максимальну безпеку мережі.

3.2.2 Атака силового повторення (Replay Attack)

Запобігання атакам Replay Attack потребує впровадження механізмів захисту в самому протоколі блокчейну.

Ось декілька способів, якими блокчейн-мережі можуть запобігти цьому виду атак:

— Використання унікальних номерів ланцюга: як вже згадувалося, Ethereum використовує унікальні номери ланцюга в своїх транзакціях, щоб забезпечити, що транзакції не можуть бути повторно використані на іншому ланцюгу. Це важливий механізм, який допомагає запобігти Replay Attacks.

— Використання Replay Protection: деякі блокчейни впроваджують вбудований механізм захисту від повторного відтворення. Наприклад, після жорсткого розколу Bitcoin і створення Bitcoin Cash в 2017 році, Bitcoin Cash впровадив Replay Protection, щоб запобігти повторному використанню транзакцій між двома ланцюгами.

— Використання нового адресу після розколу: ще одним простим, але ефективним способом запобігти Replay Attacks є переключення на новий адресу після розколу блокчейну. Це гарантує, що всі майбутні транзакції не можуть бути повторно використані, оскільки новий адреса не існує в оригінальному ланцюгу.

— Спеціалізоване програмне забезпечення: є спеціалізовані програмні інструменти, які можуть допомогти користувачам уникнути Replay Attacks. Ці інструменти включають в себе різні механізми для захисту від повторного використання транзакцій.

У контексті блокчейн-мереж, Replay Attacks становлять відчутну небезпеку, найвиразніше проявляючись під час подій розгалуження. Але з використанням виважених і ефективних стратегій захисту, цей ризик може бути істотно обмежений.

3.2.3 Атака Sybil

Атака Sybil – це ситуація, коли один користувач створює багато вузлів у мережі і представляє себе як багато різних учасників. Це може дозволити

зловмиснику отримати непропорційний вплив на мережу, а також допустити підроблення або викривлення інформації, що передається через мережу.

У системі без авторизації, такій як більшість блокчейн-мереж, кожний вузол може мати однакові права в мережі. Наприклад, в системі, яка використовує алгоритм Proof of Stake для досягнення консенсусу, користувач, який контролює велику кількість вузлів, може отримати перевагу над іншими учасниками. Це може призвести до маніпуляції результатами голосування або навіть до контролю над рішеннями мережі.

Захист від атак Sybil часто зводиться до того, щоб зробити дорогою або неможливою реєстрацію багатьох вузлів одним користувачем. Декілька можливих підходів до вирішення цієї проблеми включають:

— Proof of Work або Proof of Stake: ці механізми дозволяють учасникам демонструвати вклад (через обчислювальну потужність або ставки) як спосіб обмежити кількість вузлів, які вони можуть контролювати. Це знижує ймовірність атаки Sybil, оскільки втручання в мережу стає дороговартісним.

— Репутаційні системи: у деяких мережах використовуються системи репутації, які нагороджують вузли за добросовісну поведінку і покарають за зловмисне поведінку. Це може змусити зловмисників витратити більше ресурсів на підробку високої репутації.

— Обмеження з'єднань: щоб запобігти використанню великої кількості вузлів одним користувачем, можна обмежити кількість з'єднань від одного IP-адреси.

— Одноразові ключі ідентифікації: деякі системи використовують одноразові ключі ідентифікації для встановлення унікальних вузлів. Це означає, що зловмисник повинен буде використовувати унікальний ключ для кожного додаткового вузла, що складає значні труднощі.

— Використання централізованих сертифікатів: це може здатися контрінтуїтивним у контексті децентралізованої технології, але в деяких

випадках може бути корисним використовувати централізовані сертифікати для валідації вузлів. Це може бути особливо корисним у приватних блокчейн-мережах, де довіра між учасниками є важливою.

— Федеративні системи: у цих системах група визначених вузлів має авторитет для перевірки транзакцій і підтвердження блоків. Це допомагає знизити ризик атак Sybil, оскільки необхідно скомпрометувати значну частину цих вузлів для успішної атаки.

Загалом, хоча атаки Sybil є серйозною загрозою для блокчейн-мереж, існує багато ефективних стратегій для їх протидії. Важливо розуміти ці загрози і застосовувати відповідні механізми безпеки при проектуванні та розгортанні блокчейн-мереж.

3.2.4 Атака Eclipse

Атака Eclipse є видом атаки на безпеку мережі, при якій зловмисник або група зловмисників намагається отримати контроль над всіма з'єднаннями вузла в мережі. Це означає, що зловмисник може ізолювати вузол від решти мережі та контролювати всю інформацію, яку цей вузол отримує та передає.

Приклад атаки Eclipse: уявімо, що є зловмисник, який хоче провести атаку Eclipse на блокчейн Bitcoin. Цей зловмисник може створити безліч вузлів у мережі Bitcoin і спробувати з'єднати їх з цільовим вузлом. Якщо зловмисник успішно встановить усі вхідні та вихідні з'єднання вузла, він може ізолювати цей вузол від решти мережі.

Тепер, коли вузол хоче транзакцію або блок, він звертається до вузлів, які контролює зловмисник. Зловмисник може використовувати цю можливість для того, щоб надавати вузлу неправильну інформацію або відмовлятись від передачі будь-якої інформації взагалі. Це може призвести до різних небезпечних наслідків, зокрема до подвійних витрат.

Вирішення проблеми:

— Обмеження кількості вхідних з'єднань: більшість систем блокчейн обмежують кількість вхідних з'єднань для вузла, що допомагає запобігти встановленню контролю над всіма з'єднаннями вузла.

— Випадкові з'єднання: замість того, щоб дозволити вузлам вибирати, з ким вони з'єднуються, мережі можуть вимагати від вузлів встановлювати випадкові з'єднання. Це збільшує важкість проведення атаки Eclipse, оскільки зловмисникам важко передбачити, з ким вузол з'єднається.

— Ротація з'єднань: вузли можуть періодично змінювати з'єднання з іншими вузлами. Це збільшує важкість проведення атаки Eclipse, оскільки зловмисник повинен постійно входити до списку з'єднань вузла.

— Застосування репутаційних систем: репутаційні системи можуть бути використані для ідентифікації та відключення підозрілих вузлів, зменшуючи шанси на успішну атаку Eclipse.

— Використання фільтрації IP-адрес: додатковий підпункт може включати використання фільтрації IP-адрес для контролю доступу до вузлів мережі блокчейну.

Важливо враховувати ці заходи безпеки при проектуванні та розгортанні блокчейн-мереж для забезпечення їх стійкості до атак Eclipse.

3.2.5 Атака на протоколи консенсусу

Протоколи консенсусу в блокчейні слугують головною механікою для досягнення узгодженості між учасниками мережі щодо стану ланцюга блоків. Вони також визначають спосіб, яким блоки додаються до блокчейна, що робить їх вразливими до ряду потенційних атак.

Прикладом такої атаки може бути атака "Long Range" (довгого діапазону), яка є особливо вразливою для протоколів на основі Proof of Stake (PoS). Цей тип атаки включає учасника, який створює альтернативну версію історії блокчейна,

відправляючи блоки здалеку в минуле. Зловмисник може "переписати" історію, включаючи в неї неправдиві транзакції, які шкодять мережі.

Рішення атаки "Long Range" включає в себе використання так званих "фіналізаторів", які фіксують певні блоки як остаточні, що не можуть бути змінені або видалені. Додатково, може бути використано механізм "перевірки участі", який вимагає від учасників мережі регулярно демонструвати свою участь у мережі, щоб утримувати свої права на внесення нових блоків.

У випадках, коли протокол консенсусу не захищено від певних видів атак, може бути необхідно переглянути протокол та внести в нього відповідні зміни, щоб він міг ефективно протистояти потенційним загрозам.

Ще одним прикладом атаки на протоколи консенсусу є "Nothing at Stake" атака. Ця атака специфічна для протоколів Proof of Stake (PoS). Проблема виникає тоді, коли блокчейн форкується на дві гілки. У мережі Proof of Work (PoW), майнери витрачають ресурси, щоб добувати блоки, тому їм вигідніше працювати лише над однією гілкою. Але в PoS, учасникам не витратно добувати блоки в обох гілках, і тому вони можуть зробити це без будь-яких наслідків. Це може привести до ситуації, коли в мережі буде кілька версій історії, що підриває довіру до самої мережі.

Рішенням для цієї проблеми може бути використання покарань для валідаторів, які добувають блоки на декількох гілках одночасно. Це може бути реалізовано за допомогою так званого механізму "Slashing Conditions", що виключає можливість безкарного добування блоків на декількох гілках.

У кінцевому підсумку, ключ до захисту протоколів консенсусу від атак полягає в поєднанні технічних рішень та економічних стимулів, щоб забезпечити безпеку і стабільність мережі блокчейна. Це включає в себе розуміння потенційних вразливостей та розробку відповідних стратегій захисту. У доповнення до технічних стратегій, існують й інші способи обмеження потенційних атак на протоколи консенсусу. Одним з них є проведення постійного моніторингу мережі. Це допомагає виявити нестандартну поведінку або підозрілі патерни, які можуть вказувати на спробу атаки.

Інший спосіб полягає в створенні прозорого і демократичного середовища для всіх учасників мережі. Наприклад, багато мереж блокчейна застосовують систему голосування для вибору валідаторів або для прийняття важливих рішень. Це допомагає впевнитися, що ніхто не має надмірної влади та зменшує ризик маніпуляцій з мережею.

Також важливо регулярно оновлювати та модернізувати протоколи консенсусу, щоб вони відповідали найновішим стандартам безпеки та технологій.

І нарешті, для ефективного протидії атакам на протоколи консенсусу важливою є співпраця з усією глобальною спільнотою блокчейна. Чим більше учасників мережі, тим важче зманіпулювати системою. Крім того, активна та розумна спільнота може допомогти виявити і вирішити потенційні проблеми набагато швидше.

3.3 Аналіз вразливостей смарт-контрактів

Смарт-контракти, як і будь-який інший програмний код, можуть бути вразливі до різноманітних атак та помилок. При цьому розуміння та аналіз цих вразливостей є важливим кроком у забезпеченні безпеки блокчейн-мережі.

Однією з найпоширеніших вразливостей смарт-контрактів є помилки у коді. Це може включати різні недоліки, такі як недостатнє обробка помилок, неоптимальне використання пам'яті та витоки даних. Такі помилки можуть використовуватися атакуючими для отримання несанкціонованого доступу до функцій контракту або для викрадення криптовалюти.

Ще одним типом вразливості є атаки на рекурсію. Вони відбуваються, коли зловмисник використовує функцію виклику для того, щоб змусити смарт-контракт виконати операції, які він не планував виконувати. Це може включати надмірне використання газу, що може призвести до відмови у обслуговуванні, або неконтрольоване збільшення балансу, що може призвести до крадіжки коштів.

Атаки на оракулів - це ще один тип вразливості смарт-контрактів. Оракули - це зовнішні сервіси, які забезпечують смарт-контракти даними з реального світу. Якщо оракул компрометується або маніпулюється, він може подавати неправильні дані, що можуть призвести до неправильної роботи смарт-контракту.

3.3.1 Зловживання цільовим кодом

Зловживання цільовим кодом в смарт-контрактах є ще одним важливим питанням, яке впливає на безпеку блокчейн-мереж. Це може відбуватися, коли зловмисники здатні використати слабкі місця в коді смарт-контракту, щоб провести шкідливі дії. Така поведінка включає, але не обмежується, проникненням в смарт-контракт, модифікацією його коду або даних, та/або запуском шкідливих функцій.

За прикладом, можна взяти випадки, коли зловмисник може внести зміни до смарт-контракту таким чином, що він зможе викрасти криптовалюту або використовувати контракт для виконання дій, які не були передбачені його початковим наміром.

Для протидії такого роду атакам, розробники повинні слідкувати за останніми рекомендаціями з безпеки смарт-контрактів, включаючи, але не обмежуючись, написанням коду, який є відкритим для аудиту, тестуванням контрактів перед розгортанням, і використанням формалізованого верифікаційного процесу для перевірки коректності коду. Велике значення для підвищення безпеки має обговорення дизайну смарт-контрактів з командою, яка включає не лише розробників, але й експертів з інших сфер, таких як безпека інформації, щоб виявити можливі слабкі місця.

Для аналізу безпеки контракту може бути використаний віртуальний аналізатор безпеки, який виконує символьне виконання коду контракту і виявляє потенційні вразливості. Такі інструменти, як Securify, Mythril або Slither, мають великий потенціал для виявлення вразливостей та внесення змін до коду.

Окрім цього, важливо виконувати процес аудиту коду смарт-контрактів. Професійний аудит коду допомагає виявити вразливості, які можуть бути пропущені під час внутрішнього тестування.

Також корисною може бути практика баг-баунті, коли організація винагороджує зовнішніх дослідників за виявлення вразливостей. Це стимулює виявлення і виправлення можливих вразливостей до того, як вони можуть бути використані зловмисниками.

Таким чином, комплексний підхід до безпеки, який включає превентивні заходи, професійний аудит, і стимулювання зовнішніх дослідників, може допомогти відчутно зменшити ризик зловживання цільовим кодом в смарт-контрактах. Окрім зазначених вище технік безпеки, добре структуроване тестування є ще одним критично важливим елементом запобігання зловживань. Це означає проведення єдиного тестування, тестування компонентів, інтеграційного тестування, а також стрес-тестування для виявлення можливих слабких місць.

Приклад зловживання цільовим кодом в контексті смарт-контрактів Ethereum: в одному із випадків зловмисники змогли використати так званий "переповнення стека", який дозволив їм створювати значну кількість токенів без відповідного зниження балансу власника. Це було здійснено шляхом використання спеціально підібраного числа, що при виконанні операції множення перевищує максимально можливе значення 256-бітового числа в Ethereum. Замість помилки, виник переповнення, і результат множення став дорівнювати нулю, що зловмисники використали для створення токенів.

Рішення цієї проблеми полягає у використанні бібліотек безпеки, таких як OpenZeppelin для Ethereum, які надають безпечні математичні функції, що гарантують виклик помилки у випадку переповнення. Це запобігає можливому зловживанню цільовим кодом шляхом використання цього типу атаки.

Всі ці методи в сукупності дозволяють знизити ризик зловживання цільовим кодом, підвищуючи безпеку смарт-контрактів на всіх етапах їхнього життєвого циклу, від розробки до розгортання і використання.

3.3.2 Помилки в коді

Можливість помилок в коді смарт-контрактів є неминучою і може призвести до вразливостей, які зловмисники можуть використовувати. Помилки в коді можуть бути результатом різних факторів, включаючи недостатнє розуміння розробниками мови програмування, спішку, недостатню відповідальність за якість коду, або просто помилки, які не були виявлені під час тестування.

Одним з найбільш відомих прикладів помилки в коді смарт-контракту був інцидент DAO (Decentralized Autonomous Organization) в 2016 році, коли зловмисники використали помилку в смарт-контракті DAO для виведення приблизно 3.6 мільйонів Ether, що на той час складало близько 60 мільйонів доларів.

Ця помилка була заснована на вразливості, відомій як "рекурсивний виклик". Зловмисник використав цю вразливість для того, щоб почати виведення коштів перед тим, як було оновлено баланс DAO. Це дозволило зловмиснику вивести більше коштів, ніж було на балансі DAO.

Щоб запобігти таким помилкам в коді, розробники повинні дотримуватися кількох основних принципів. По-перше, вони повинні повністю розуміти мову програмування, на якій вони працюють, і її специфіку в контексті смарт-контрактів. По-друге, вони повинні проводити ретельне тестування та аудит свого коду перед його використанням. Використання формалізованих методів верифікації може бути корисним для забезпечення високої якості коду. Крім того, використання безпечних шаблонів кодування та бібліотек, як-то OpenZeppelin в Ethereum, може допомогти в запобіганні поширених помилок.

Важливо зауважити, що, незважаючи на всі зусилля, повністю виключити можливість помилок в коді неможливо. Однак, дотримання найкращих практик розробки та тестування може значно знизити їх вплив і ризик експлуатації.

3.3.3 Відмова в роботі

Відмова в роботі (DoS атака) - це стратегія зловмисника, при якій він намагається зробити ресурс недоступним для його призначеного використання. В контексті блокчейну та смарт-контрактів, DoS атака може призвести до перевантаження системи, що в подальшому призводить до підвищення часу обробки транзакцій та навіть до їх повної зупинки.

Смарт-контракти, розроблені на Ethereum, особливо вразливі до DoS атак. За виключенням атаки на весь блокчейн, існують два основних типи атак DoS, які зосереджуються на смарт-контрактах: атаки на газ і атаки на великі кількості керування. У смарт-контрактах використовується поняття "газу" для вимірювання обчислювальних ресурсів, необхідних для виконання операцій.

Атаки на газ стосуються витрат газу в Ethereum. Кожна транзакція в мережі Ethereum вимагає певної кількості газу, який сплачується в Ether. Зловмисники можуть створювати транзакції, що витрачають велику кількість газу, щоб сповільнити обробку інших транзакцій.

Атаки на великі кількості керування зосереджуються на використанні динамічних структур даних у смарт-контрактах, які можуть стати великими до такої міри, що вони будуть вимагати велику кількість газу для обробки. Це може заблокувати функціонування смарт-контракту.

Рішення проблеми:

— Застосування границь на транзакції або витрати газу.

— Розробка смарт-контрактів з урахуванням величин структур даних.

— Модифікація коду смарт-контракту, щоб зменшити витрати газу.

— Використання механізмів оцінки витрат газу перед виконанням транзакцій.

— Планування використання ресурсів і встановлення максимальних лімітів на витрати газу.

Одним із способів боротьби з DoS-атаками є використання гідрачних контрактів, які дозволяють розподіляти обчислювальне навантаження між декількома контрактами. Кожен з цих контрактів може обробляти лише частину загальних даних, що дозволяє уникнути перевантаження одного контракту. Це особливо корисно при великому обсязі даних, які потребують обробки.

Також, ефективним рішенням є впровадження моделі "платіж за користування". Вона полягає в тому, що користувач, який виконує транзакцію, сплачує комісію, пропорційну розміру даної транзакції або кількості використовуваного для її обробки ресурсу. Це зменшує економічну вигоду для зловмисника від проведення DoS-атаки, оскільки вона стає занадто дорогою.

Нарешті, створення та впровадження ефективних стратегій моніторингу та виявлення аномалій також може допомогти у виявленні та запобіганні DoS-атак. Ці стратегії можуть включати в себе використання системи детектування вторгнень (IDS), аналіз трафіку та поведінки користувачів для виявлення аномальної активності, та встановлення системи алертування для швидкого реагування на можливі DoS-атаки.

3.3.4 Оракул-атаки

Оракул-атака є специфічною для смарт-контрактів, що потребують зовнішнього входу даних від "оракулів". Оракули в контексті смарт-контрактів – це треті сторони, які забезпечують смарт-контракти необхідною інформацією ззовні блокчейну, наприклад, актуальними валютними курсами, цінами на товари, погодними даними та іншою інформацією.

Суть оракул-атаки полягає в маніпуляції цими зовнішніми даними. Наприклад, зловмисник може використати компрометований оракул для надсилання неправильних даних до смарт-контракту, що може призвести до втрати коштів або інших небажаних результатів.

Прикладом оракул-атаки може бути випадок, коли смарт-контракт, розроблений для автоматизованого страхування погодних ризиків, базує свої

рішення на даних від певного оракула, що надає інформацію про погодні умови. Якщо зловмисник компрометує оракул і викривлює дані про погоду, смарт-контракт може неправильно виплатити страхові виплати, що призведе до великих фінансових втрат.

Для запобігання оракул-атакам можна використовувати декілька методів. По-перше, важливо вибирати надійних постачальників оракулів. Вони повинні мати хорошу репутацію, бути транспарентними і використовувати безпечні методи для передачі даних.

По-друге, варто використовувати кілька оракулів для забезпечення редундантності і диверсифікації ризику. Це дозволяє зробити процес взяття рішень більш надійним, оскільки навіть якщо один оракул стає компрометованим, інші можуть надати правильні дані.

Також, можна розглянути використання децентралізованих оракулів, які використовують консенсус між кількома незалежними вузлами для визначення правдивості даних. Це може додатково знизити ризик маніпуляції даними.

3.4 Реалізація алгоритму PoS для запобігання атаки 51%

Реалізація алгоритму PoS, що може бути використаний для запобігання атакам 51%, зображено на рисунку 3.5. Як видно, спочатку створено нові блоки в ланцюгу з використанням функції `new_block()`. Кожен блок містить транзакції, що чекають на обробку (зберігаються в `current_transactions`), а також відбиток попереднього блоку (`previous_hash`).

Алгоритм використовує "доказ ставки" (PoS) замість "доказу роботи" (PoW) для вирішення конфліктів при виборі, куди додати новий блок. Така схема значно знижує ризик атаки 51%, оскільки зловмисник потребує контролювати більше половини всіх токенів в мережі, а не обчислювальної потужності.

Всі розглянуті стратегії мають свої переваги та недоліки, але всі вони мають одну загальну мету: забезпечити безпеку та децентралізацію блокчейн-

мережі. Комбінація цих стратегій може привести до створення більш надійної та стабільної мережі.

Успіх цих методів, однак, значною мірою залежить від розуміння та активної участі спільноти. Отже, освіта та залучення користувачів до участі в управлінні мережею є важливими компонентами в посиленні безпеки блокчейну.

```python
class Blockchain:

    def __init__(self):
        self.chain = []
        self.current_transactions = []
        self.nodes = set()
        self.new_block(previous_hash=1, proof=100)

    def new_block(self, proof, previous_hash=None):
        block = {
            'index': len(self.chain) + 1,
            'timestamp': time(),
            'transactions': self.current_transactions,
            'proof': proof,
            'previous_hash': previous_hash or self.hash(self
                .chain[-1]),
        }
        self.current_transactions = []
        self.chain.append(block)
        return block

    def new_transaction(self, sender, recipient, amount):
        self.current_transactions.append({
            'sender': sender,
            'recipient': recipient,
            'amount': amount,
        })
        return self.last_block['index'] + 1

    @staticmethod
    def hash(block):
        block_string = json.dumps(block, sort_keys=True).encode()
        return hashlib.sha256(block_string).hexdigest()

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

Рисунок 3.5 – Реалізація алгоритму PoS

Важливо також зауважити, що з впровадженням нових технологій та прогресом в області криптографії з'являються нові можливості для підвищення безпеки блокчейн-мереж.

В підсумку, атака 51% є реальним загрозою для блокчейн-мереж, особливо для тих, що базуються на PoW. Проте, є різні стратегії для підвищення безпеки та зменшення ймовірності такої атаки. Ефективне використання цих стратегій може забезпечити надійність та децентралізацію, що є основними цілями блокчейн-технології.

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Долікарська допомога при харчових отруєннях

4.1.1 Ознаки та симптоми харчових отруєнь

Харчові отруєння зазвичай виникають через вживання забруднених або несвіжих продуктів. Патогени або їх токсини, які присутні в харчових продуктах, можуть викликати шкідливі реакції організму. Симптоми можуть проявитися від декількох годин до декількох днів після вживання забрудненої їжі або води. Основні ознаки та симптоми харчових отруєнь включають:

— нудота і блювання: ці симптоми є дуже поширеними при харчових отруєннях. Організм спробує вивести токсини шляхом викликання блювотних рефлексів;

— діарея: при харчових отруєннях зазвичай виникає рідкий або водянистий стілець. Це ще один спосіб, яким організм намагається вивести токсини;

— болі в животі та кишечнику: болі можуть бути гострими та крампоподібними і зазвичай виникають через запалення або подразнення кишечника;

— головний біль: спостерігається внаслідок загального впливу токсинів на організм;

— підвищення температури тіла: відповідь організму на інфекційний агент;

— слабкість, втрата апетиту, запаморочення: ці симптоми можуть виникнути внаслідок дегідратації, викликаної постійним блюванням і діареєю.

У важких випадках харчового отруєння можуть виникнути більш серйозні симптоми, такі як кров у стільці, висока температура, судоми, дезорієнтація або

навіть обморок. Ці симптоми вимагають невідкладної медичної допомоги.

4.1.2 Принципи надання першої долікарської допомоги при харчових отруєннях

Харчові отруєння можуть бути дуже неприємними та потенційно небезпечними. Раннє виявлення симптомів і надання ефективної першої допомоги може суттєво полегшити стан хворого та сприяти швидшому відновленню. Основні принципи надання першої допомоги при харчових отруєннях включають:

— припинення вживання продуктів: якщо є підозра на харчове отруєння, то подальше вживання підозрілого продукту або їжі повинне бути негайно припинено, щоб запобігти подальшому вживанню потенційно шкідливих речовин;

— забезпечення дехідратації: харчові отруєння часто супроводжуються блюванням та діареєю, що може викликати втрату води та важливих електролітів. Прийом великої кількості чистої води або розчинів регідратації може допомогти поповнити втрачені рідини;

— викликати швидку: у випадках серйозного харчового отруєння, коли симптоми включають високу температуру, судоми, обморок або кров'яні діареї, необхідно негайно викликати швидку допомогу;

— моніторинг стану: постійний моніторинг стану хворого дозволяє вчасно виявити погіршення стану здоров'я та прийняти відповідні дії;

— обережне прийняття їжі та пиття: після впевненості в стабілізації стану, поступово можна вводити легку їжу та пиття, таку як бульйони, прісний рис, банани та яблучний сік. Це допоможе запобігти подальшій дехідратації та стабілізувати рівень цукру в крові;

— покращення харчової гігієни: для запобігання подальшим випадкам харчового отруєння, важливо дотримуватись належної харчової гігієни. Це включає ретельне миття рук перед приготуванням їжі, використання окремих дошок та ножів для м'яса та овочів, термінове зберігання їжі в холодильнику та вживання свіжих продуктів.

4.1.3 Медична допомога при харчових отруєннях

Після надання першої допомоги, особа з симптомами харчового отруєння має звернутися до медичного працівника для подальшого огляду та лікування. Медична допомога при харчових отруєннях включає наступні етапи:

— діагностика: при першому огляді лікар зазвичай збирає анамнез, включаючи інформацію про їжу та воду, що були вжиті, час виникнення симптомів та їх перебіг. За необхідності можуть бути використані лабораторні тести (наприклад, аналіз стільця) для визначення конкретного патогена;

— лікування: в залежності від симптомів та викликаного патогена, лікування може включати відновлення рівня електролітів і гідратації, вживання пробіотиків для відновлення здорової мікрофлори кишечника, антібіотики для бактеріальних інфекцій та антиеметики для контролю нудоти та блювання;

— надання рекомендацій: лікар може надати поради щодо дієти та режиму пиття в період відновлення. Це може включати вживання легких бульйонів, прісного рису, бананів, яблучного соку, тостів та інших легко засвоюваних продуктів;

— моніторинг: подальший моніторинг стану пацієнта важливий для контролю процесу відновлення та виявлення можливих ускладнень. Це може включати додаткові візити до лікаря або консультації;

— госпіталізація: у випадках важкого харчового отруєння, коли домашнє лікування недостатньо або є ризик ускладнень, може бути потрібна

госпіталізація для інтенсивного лікування та нагляду;

— інформування органів здоров'я: якщо є підозра на масовий випадок харчового отруєння, важливо повідомити місцеві органи здоров'я, такі як санепідслужбу або місцеву лікарню. Це допоможе сприяти виявленню та контролю поширення інфекції, а також може сприяти вжиттю необхідних заходів для запобігання подальшим випадкам харчового отруєння у громаді.

4.1.4 Профілактика харчових отруєнь

Профілактика є критично важливим аспектом управління харчовими отруєннями. Вона включає ряд дій, які спрямовані на запобігання виникнення харчових отруєнь за рахунок виключення або зменшення ризику потрапляння патогенів у продукти харчування. Основні принципи профілактики харчових отруєнь включають:

— правильне зберігання продуктів харчування: продукти, особливо швидкопсувні, повинні зберігатися при відповідних температурах, щоб запобігти розмноженню патогенних мікроорганізмів;

— виконання санітарних норм при приготуванні їжі: руки та кухонне обладнання повинні бути чистими. Сировина повинна бути належно відділена від готової їжі, щоб запобігти перехресному забрудненню;

— правильна термічна обробка продуктів: продукти, особливо м'ясо та яйця, повинні бути достатньо приготовані, щоб вбити потенційні патогени;

— вживання безпечної питної води: вода може бути джерелом патогенів, особливо в регіонах з поганими санітарними умовами. Вживання кип'яченої або бутильованої води може допомогти уникнути харчових отруєнь;

— освіта та інформування: знання про правильне приготування та зберігання продуктів, а також здатність розпізнати симптоми харчового

отруєння, є важливими інструментами для профілактики;

— покращення гігієни закладів харчування: ресторани, кафе, їдальні та інші заклади харчування повинні дотримуватись високих стандартів санітарії та гігієни.

Профілактика харчових отруєнь вимагає зусиль на різних рівнях - від виробників і постачальників продуктів до кінцевих споживачів. Всі мають відповідальність за забезпечення безпеки харчових продуктів;

4.1.5 Як блокчейн може допомогти в сфері харчових отруєнь

Блокчейн може допомогти в сфері харчових отруєнь, зокрема, за допомогою забезпечення прозорості та слідуваності у ланцюгах поставок харчових продуктів. Це може бути особливо корисним при розслідуванні причин харчових отруєнь та при профілактиці подібних випадків в майбутньому.

— прозорість у ланцюгу поставок: блокчейн може відслідковувати та записувати всі етапи переміщення продуктів від ферми до столу. Це означає, що у випадку харчового отруєння, організації можуть швидко виявити джерело проблеми, зрозуміти, де було введено шкідливі речовини або де сталися порушення санітарних норм;

— швидкість розслідування: традиційно, виявлення джерела харчового отруєння може зайняти велику кількість часу. Блокчейн дозволяє значно пришвидшити цей процес, допомагаючи знайти початкове джерело проблеми в ланцюгу поставок;

— попередження харчових отруєнь: з використанням блокчейна, підприємства можуть запобігти потенційним проблемам з безпекою харчових продуктів, виявляючи та усуваючи проблеми на ранніх етапах ланцюга поставок;

— забезпечення довіри споживачів: за допомогою блокчейн-технології, споживачі можуть отримувати достовірну та перевірену інформацію про

походження та якість харчових продуктів. Це дозволяє збільшити довіру до брендів і виробників, зменшити ризик харчових отруєнь і покращити споживчий досвід.

Використання технології блокчейн у ланцюгах поставок харчових продуктів все ще знаходиться на початкових стадіях, але вже має великий потенціал для покращення безпеки харчових продуктів та зменшення ризику харчових отруєнь.

4.2 Заходи щодо автоматизації виробничих процесів, які сприяють покращенню умов праці

Автоматизація виробничих процесів стає все більш поширеною у різних галузях промисловості. Це не тільки підвищує ефективність та продуктивність, але також може значно покращити умови праці, зменшуючи фізичні навантаження та ризики для працівників.

4.2.1 Автоматизація важких та небезпечних задач

Автоматизація може відіграти критичну роль у віддаленні працівників від важких та небезпечних задач, що може покращити умови праці та зменшити ризик робочих травм. Ось декілька способів, якими автоматизація може бути використана в цьому контексті:

— автоматизація важких фізичних завдань: машини та роботи можуть виконувати роботу, яка зазвичай вимагає від людей великих фізичних зусиль. Наприклад, підйом важких вантажів, транспортування товарів та матеріалів, тощо. Використання машин знижує ризик пов'язаних з цим травм та хронічних захворювань опорно-рухового апарату;

— автоматизація небезпечних задач: роботи можуть виконувати завдання у небезпечних умовах, де є ризик вибуху, пожежі, випромінювання або

хімічного отруєння. Це забезпечує безпеку працівників, віддаляючи їх від безпосереднього контакту з небезпечним середовищем;

— автоматизація роботи в високо- або низькотемпературних умовах: роботи можуть працювати в умовах, які є занадто жаркими або холодними для людей, що знижує ризик теплових або холодових травм;

— автоматизація монотонних задач: роботи також можуть виконувати рутинні, повторювані завдання, що в подоланні монотонності роботи та стресу, асоційованого з цим.

Таким чином, автоматизація значно зменшить ризик професійних травм та забезпечити безпечніші умови праці. Проте, при впровадженні автоматизованих систем, враховувють й потенційні нові ризики, такі як несправності обладнання, та розробити відповідні заходи безпеки.

4.2.2 Покращення точності та консистентності

Автоматизація виробничих процесів може значно покращити точність та консистентність виробництва, що, в свою чергу, підвищує якість продукції та забезпечує безпеку працівників. Ось декілька аспектів цього питання:

— покращення точності: машини та роботи здатні виконувати складні завдання з великою точністю, що може зменшити кількість помилок у процесі виробництва. Це особливо важливо у сферах, де помилки можуть призвести до серйозних наслідків в авіації, медицині, хімічній промисловості;

— покращення консистентності: роботи можуть виконувати однакові завдання безперервно без втрати якості або продуктивності. Це гарантує стабільність виробничого процесу і дозволяє виробляти продукцію з постійним рівнем якості;

— зниження ризику помилок та нещасних випадків: за допомогою автоматизації можна зменшити кількість помилок, зроблених через людський

фактор, що, в свою чергу, зменшує ризик нещасних випадків на виробництві;

— забезпечення неперервності процесів: автоматизовані системи можуть працювати 24/7 без втрати продуктивності, що забезпечує неперервність виробничих процесів і зменшує тиск на працівників.

Враховуючи ці фактори, робимо висновок, що автоматизація значно покращить умови праці шляхом покращення точності та консистентності виробничих процесів.

4.2.3 Використання блокчейн в заходах щодо автоматизації виробничих процесів, які сприяють покращенню умов праці

Блокчейн може використовуватись для автоматизації виробничих процесів з метою покращення умов праці наступними способами:

— смарт-контракти: це автоматизовані контракти, що самі виконують певні дії при виконанні визначених умов. Вони можуть використовуватися для автоматизації різних виробничих процесів, зменшуючи потребу в ручному управлінні та знижуючи ризик помилок;

— підтвердження праці: блокчейн може служити незмінним журналом дій, що виконуються на виробництві. Це може зменшити потребу в постійному нагляді за працівниками та знизити стресові навантаження.

— безпека даних: блокчейн може гарантувати безпеку виробничих даних і забезпечити їх від неправомірного доступу. Це дозволяє працівникам працювати в безпечній обстановці, знаючи, що їхні дані захищені;

— журнал виконання роботи: блокчейн може бути використаний для ведення точного журналу виконання роботи, який автоматично відслідковує час роботи, виконані задачі та інші важливі аспекти роботи. Це може допомогти зменшити надмірну роботу і втому;

— покращення прозорості та відслідковуваності: блокчейн може допомогти автоматизувати процеси звітності та відслідковування виробничих процесів, що забезпечує більшу прозорість та може підвищити безпеку та ефективність на робочому місці.

Узагальнюючи, блокчейн може використовуватись для автоматизації виробничих процесів з метою покращення умов праці, зниження стресу та втоми серед працівників, підвищення ефективності та безпеки на виробництві.

ВИСНОВКИ

В результаті в роботі проаналізовано принцип роботи та види вразливостей технології блокчейн. Виявлено, що блокчейн-програми виходять далеко за межі криптовалюти та біткойнів. Завдяки здатності створювати більшу прозорість і справедливість, а також економити час і гроші бізнесу, ця технологія впливає на різноманітні сектори різними способами: від виконання контрактів до підвищення ефективності роботи уряду. Фахівці стверджують, що blockchain-транзакції будуть покладені в основу нового покоління Інтернету Web3. В ідеалі ця мережа повинна ставити на перший план користувача — її хочуть зробити вільною та децентралізованою, з мінімальним рівнем цензури, монополій та державного втручання. У підсумку, підтримка безпеки та стійкості блокчейн-мережі є багатоаспектним завданням, яке вимагає комплексного підходу та співпраці усіх учасників екосистеми. Тільки таким чином можна досягти довгострокової стабільності та забезпечити ефективну роботу мережі в умовах постійно змінного кіберпростору.

В першому розділі було визначено основні поняття технології блокчейну та її принципи роботи. Були проаналізовані протоколи консенсусу, які використовуються в блокчейні, а також технологічні та організаційні методи підвищення безпеки блокчейн-мереж.

Другий розділ присвячений технологічним та організаційним методам підвищення безпеки блокчейн-мереж. Зокрема, було здійснено аналіз застосування хеш-функцій для підвищення криптостійкості, вивчено алгоритми MD5 та SHA-1, а також проведено аналіз методів підвищення криптографічної стійкості.

У третьому розділі були вивчені алгоритми шифрування для підвищення безпеки блокчейн-мереж, а також проведено аналіз основних типів вразливостей блокчейн-мереж і типових атак на них. Було проведено детальний аналіз атаки 51%, а також розглянуто можливі методи запобігання

цьому виду атак на підставі реалізації алгоритму PoS. Окрім того, були проаналізовані основні вразливості смарт-контрактів.

Загалом, дана робота дає цілісне розуміння проблем безпеки в блокчейн-мережах і можливих шляхів їхнього вирішення. Було зроблено важливий внесок у вивчення способів підвищення криптостійкості блокчейну та аналізу потенційних вразливостей. Отримані результати мають важливе теоретичне та практичне значення, і можуть бути використані при розробці нових стратегій збільшення безпеки блокчейн-мереж.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. І.В. Лисенко, Порівняльна характеристика можливостей програмних платформ і мов програмування з точки зору реалізації криптоалгоритмів / І.В. Лисенко, Ю.В. Трегуб - Системи управління, навігації та зв'язку. – 2017- випуск 1(41).

2. Stallings W. Cryptography and network security principles and practices // International Journal - Pearson Education, Inc, 2016 – 4 th edition – p.268.

3. Основи технології блокчейн: комп'ютерний практикум [Електронний ресурс]: навч. посіб. для студентів спеціальностей 126 «Інформаційні системи та технології» та 121 «Інженерія програмного забезпечення» / КПІ ім. Ігоря Сікорського; уклад.: В.А. Яланецький. – Електронні текстові дані (1 файл: 2.1 Мбайт). – Київ: КПІ ім. Ігоря Сікорського, 2022. – 89 с. https://ela.kpi.ua/bitstream/123456789/47876/1/Osnovy.pdf

4. Довжик Д.В. (2019) Система онлайн-голосування на базі технології Blockchain (магістерська дисертація) . НТУУ «КПІ» ім. І. Сікорського https://ela.kpi.ua/bitstream/123456789/38416/1/Dovzhyk_magistr.pdf

5. Van Flymen, Daniel. 2020. Learn Blockchain by Building One : A Concise Path to Understanding Cryptocurrencies. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-5171-3.

6. М о р гу н О .М . К р іп то гр аф іч н і м ето д и зах и сту ін ф орм ац ії: Н авч. посібни к. - Ч еркаси : А П Б ім. Г ер о їв Ч о р н о б и л я, 2008. - 97с.

7. T. Ahram, A.Sargolzaei, S.Sargolzaei, J.Daniels, B. Amaba. “Blockchain technology innovations,” in Proc. Technology & Engineering Management (TEMSCON), 2017 IEEE Conference on, 2017, P. 137–141.

8. “Proof of Stake versus Proof of Work”. White Paper, Internet: https://bitfury.com/content/downloads/pos-vs-pow-1.0.2.pdf Jan. 22, 2019.

9. N. Abdullah, A. Hakansson, E. Moradian. “Blockchain based approach to enhance big data authentication in distributed environment,” in Proc. 9th International Conf. Ubiquitous and Future Networks (ICUFN), IEEE, 2017, P. 887-892.

10. Satoshi Nakamoto. “Bitcoin: A Peer-to-Peer Electronic Cash System”. Internet: https://bitcoin.org/bitcoin.pdf Dec. 12, 2018.

11. Z. Zheng, S.Xie, H. N.Dai, H.Wang. “Blockchain challenges and opportunities: A survey”. Int. J. Web and Grid Services, vol. 14 (4), P.352-375, 2018.

12. Princeton university. Bitcoin and Cryptocurrency Technologies. https://www.coursera.org/learn/cryptocurrency July 30, 201.

ДОДАТОК А Код запобігання атаки 51%

```
class Blockchain:
def __init__(self):
self.chain = []
 self.current_transactions = []
 self.nodes = set()

 def add_node(self, address):
self.nodes.add(address)

def valid_chain(self, chain):
last_block = chain[0]
 current_index = 1

 while current_index < len(chain):
 block = chain[current_index]
if block['previous_hash'] != self.hash(last_block):
 return False

if not self.valid_proof(last_block['proof'],
block['proof']):
 return False

last_block = block
current_index += 1

return True

def resolve_conflicts(self):
neighbours = self.nodes
new_chain = None

max_length = len(self.chain)
```

```
        for node in neighbours:
        response = requests.get(f'http://{node}/chain')
        if response.status_code == 200:
        length = response.json()['length']
        chain = response.json()['chain']
        if   length   >   max_length   and   self.valid_chain(chain):
max_length = length
        new_chain = chain

        if new_chain:
        self.chain = new_chain
        return True

        return                                                    False



class Blockchain:

         def __init__(self): self.chain = []
        self.current_transactions = []
         self.nodes = set()
        self.new_block(previous_hash=1, proof=100)

         def new_block(self, proof, previous_hash=None):
         block = {
        'index': len(self.chain) + 1,
        'timestamp': time(),
        'transactions': self.current_transactions,
        'proof': proof,
        'previous_hash': previous_hash or self.hash(self.chain[-1]),
        }
        self.current_transactions = []
 self.chain.append(block)
 return block
```

```
def new_transaction(self, sender, recipient, amount):
self.current_transactions.append({
'sender': sender,
 'recipient': recipient,
'amount': amount,
 })
return self.last_block['index'] + 1

 @staticmethod
def hash(block):
block_string = json.dumps(block, sort_keys=True).encode() r
eturn hashlib.sha256(block_string).hexdigest()

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