Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Безпеки інформаційних технологій
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

магістр
(освітньо-кваліфікаційний рівень)

ГЮІК.ХХХХХ1649Ст.06ПЗ
(позначення документа)

Технологія блокчейн для побудови децентралізованої інфраструктури відкритих ключів
(тема)

Виконав: Курбатов О.С.
(прізвище, ініціали)
студент 2 курсу, групи БІКСм-18-1
Спеціальність 125 Кібербезпека
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма «Безпека інформаційних і комунікаційних систем»
(повна назва освітньої програми)
Керівник Горбенко І.Д.
(прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Халімов Г.З.
(підпис) (прізвище, ініціали)

2019 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Безпеки інформаційних технологій

Рівень вищої освіти другий (магістерський)

Спеціальність 125 Кібербезпека
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна, або освітньо-наукова)

Освітня програма «Безпека інформаційних і комунікаційних систем»
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ АТЕСТАЦІЙНУ РОБОТУ

студентові *Курбатову Олександру Сергійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) *Технологія блокчейн для побудови децентралізованої інфраструктури відкритих ключів*
затверджена наказом по університету від "04" листопада 2019 р. № 1649Ст
2. Термін подання студентом роботи до атестаційної комісії _____
3. Вихідні дані до роботи (проекту) *Теоретичні дані щодо побудови ІВК згідно Х.509*
4. Зміст пояснювальної записки (перелік питань, що потрібно розробити)
   1.Аналіз існуючих рішень та патентів, що стосуються децентралізованої ІВК
   2.Опис архітектури децентралізованої систем ідентифікації
   3.Дослідження особливостей геш-функцій, що можуть використовуватися в системі
   4. Результати досліджень
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів) *Презентаційний матеріал у вигляді слайдів*
6. Основна література та джерела
Горбенко Ю.І. Інфраструктури відкритих ключів. Електронний цифровий підпис. Теорія та практика : монографія / Ю.І. Горбенко, І.Д. Горбенко ; Харк. нац. ун-т радіоелектрон., ЗАТ Інт інформ. технологій. – Х. : Форт, 2010. – 593 с

7. Консультанти розділів роботи (проекту)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
1 Існуючі системи управління ідентифікаційними даними	Доцент каф. БІСТ ХНУ ім В.Н. Каразіна Полуяненко М.О.		
2 Архітектура децентралізованої системи ідентифікації та інфраструктури відкритих ключів	Доцент каф. БІСТ ХНУ ім В.Н. Каразіна Полуяненко М.О.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської атестаційної роботи	Термін виконання етапів роботи	Примітка
1	*Затвердження теми роботи та узгодження плану*		
2	*Дослідження джерел та наукової літератури*		
3	*Написання та представлення I та II розділу роботи*		
4	*Написання та представлення III розділу роботи*		
5	*Оформлення пояснювальної записки магістерської роботи*		
6	*Підготовка та отримання необхідних документів (відгук, зовнішня рецензія та ін.)*		

Студент _____

(підпис)

Керівник роботи (проекту) _____ д.т.н., проф. Горбенко І. Д._____

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до магістерської атестаційної роботи містить 72 ст., 16 рисунків, 6 таблиць, 75 джерел, 1 додаток.

Метою магістерської роботи є розробка архітектури децентралізованої інфраструктури відкритих ключів укупі з системою цифрової ідентифікації; початкове дослідження ефективності розробленої системи та порівняння з існуючими та запропонованими рішеннями. Об'єктом роботи є процес побудови децентралізованої інформатори відкритих ключів. У якості експериментальної частини роботи проводиться дослідження геш-функцій на предмет можливості їх використання для побудування дерев Меркла з частин персональних даних користувачів – одного зі складових механізмів запропонованої системи. Для дослідження геш-функцій розроблена програма, що дозволяє побудувати дерева Меркла за допомогою різних геш-функцій, та окремі функції, що моделюють поведінку атакуючої сторони – використовуючи частини доказів намагаються відновити вихідний набор даних.

За результатами роботи побудована відповідна архітектура ІВК та системи ідентифікації, що дозволяє забезпечити гнучкий захист від атак «людина посередині» та дозволяє кінцевим користувачам незалежно приймати рішення щодо валідності окремих сертифікатів та ідентифікаторів. Також сформовано вимоги до геш-функцій, що можуть використовуватися у описаній архітектурі. Наступним кроком дослідження може виступати аналіз інших елементів системи.

Ключові слова: ІДЕНТИФІКАЦІЯ, ІНФРАСТРУКТУРА ВІДКРИТИХ КЛЮЧІВ, БЛОКЧЕЙН, ДЕЦЕНТРАЛІЗОВАНА СИСТЕМА, ГЕШ-ФУНКЦІЯ.

ABSTRACT

Explanatory note to the master's appraisal work includes 72 pages, 16 figures, 6 tables, 75 sources, 1 appendix.

The purpose of the master's thesis is to develop an architecture of decentralized public key infrastructure together with a digital identity system; initial discovery of the effectiveness of the developed system and comparison with existing and proposed solutions. The object of the work is the process of building a decentralized public key informant. As an experimental part of the study, the study of hash functions is conducting to determine whether they could be used to construct Merkle trees from parts of users' personal data – one of the components of the proposed system. To investigate the hash functions, a program has been developed to build Merkle trees using different hash functions, and individual features that model the behavior of the attacker – using pieces of evidence to try to reconstruct the original data set.

As a result of the work, an appropriate PKI architecture and identity system was built to allow flexible protection against “man-in-the-middle” attacks and allow end-users to make independent decisions about the validity of individual certificates and identifiers. Also, requirements for the hash functions that can be used in the described architecture have been generated. The next step in the investigation may be to analyze other elements of the system.

Key words: IDENTITY, PUBLIC KEY INFRASTRUCTURE, BLOCKCHAIN, DECENTRALIZED SYSTEM, HASH-FUNCTION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ІВК (PKI) – інфраструктура відкритих ключів

PoW – proof of work

PoS – proof of stake

BFT – byzantine fault tolerance

FBA – federated byzantine agreement

IF – identity federation

ЦС (CA) – центр сертифікації

ПД – персональні дані

ПІП – постачальник ідентифікаційних послуг

SP – services provider

P2P – peer-to-peer

БЧ – блокчейн

SSO – single sign-on

ІД – ідентифікаційні дані

Identity – ідентифікаційний обліковий запис

ВСТУП

Існуючі інфраструктури відкритих ключів стали першим кроком у оцифровці взаємодії між користувачами Інтернету, що передбачало надання основних послуг інформаційної безпеки [1-4]. Вони показали, що ця взаємодія може бути здійснена за допомогою криптографічних методів на основі сертифікатів відкритих ключів користувачів [5-6]. Однак такі підходи все ще дуже громіздкі для звичайного користувача.

Окрім того, класична інфраструктура відкритих ключів, що базується на стандарті X.509 має ряд недоліків [7-11]. Перший з недоліків – єдина точка відмови. Центри сертифікації несуть повну відповідальність за видання, оновлення та відкликання сертифікатів користувачів. При цьому, якщо був скомпрометований центр сертифікації, то сертифікати кінцевих користувачів також фактично вважаються скомпрометованими – тобто система (або підсистема) втрачає здатність виконувати свої функції. І чим вище по ієрархії знаходиться скомпрометований центр, тим більша частина глобальної системи відмовляє у обслуговуванні [8, 10].

Другим недоліком є, у значній мірі, необхідність довіри окремому центру сертифікації (ЦС). Так як згідно діючому законодавству (та врешті і у середовищі, що склалося) передбачена генерація (та часткове управління) центром сертифікації ключів кінцевих користувачів, то така особливість визначає повну довіру по відношенню до центрів сертифікації [12-13].

Наступний недолік – проблеми синхронізації між центрами сертифікації. Кожен з центрів фактично не знає про сертифікати, видані іншим центром, що може призвести до фактичної підміни сертифікату одним з ЦС [9]. Більш того, така нерівномірність системи дуже сильно впливає на зручність використання сертифікату користувачем [8].

Наразі існує велика кількість запропонованих рішень, що передбачають просто використання технології блокчейн поверх класичної архітектури [14-15]. Такі підходи можуть вирішити низку проблем, але при цьому все ще залишається залежність від центрів сертифікації вищого рівня [16].

На даний момент обрана тема є достатньо актуально, так як зараз ми бачимо трансформацію у світі цифрових відносин. Традиційні схеми автентифікації, що використовують підхід “логін-пароль”, замінюються на системи, що потребують ідентифікатору та значення цифрового підпису для аутентифікації та авторизації користувачів. В таких умовах дуже важливе функціонування надійної інфраструктури відкритих ключів разом з системою ідентифікації. На даний момент видано близько двох десятків патентів, що присвячені цим темам (жодного старше 2010 року) [17], ведучі компанії, такі як IBM, Microsoft, Amazon та ін. намагаються побудувати такі інфраструктури у межах власних систем та сервісів, але глобальна реалізація поки що відсутня [18-20].

До цієї роботи ми вже запропонували концепцію децентралізованої інфраструктури відкритих ключів, що здатна достатньо ефективно працювати за умов невеликої кількості валідаторів [21]. Наступні кроки будуть пов’язані з вирішенням питань сумісності системи ідентифікації з сервісами, що потребують ідентифікації: платформи голосування, окремі реєстри, тощо [22].

Впровадження системи глобальної ідентичності (та інфраструктури відкритих ключів), описаної в роботі, не означає повної відмови від існуючої інфраструктури X.509. Більш того, передбачається проста інтеграція існуючих сертифікатів шляхом додавання їх до структури аккаунтів. Ключова ідея такої системи полягає саме в передачі контролю даних в руки їх власників та об'єднанні доступу до різних служб за допомогою криптографічних методів.

1 ІСНУЮЧІ СИСТЕМИ УПРАВЛІННЯ ІДЕНТИФІКАЦІЙНИМИ ДАНИМИ

На сьогоднішній день звичним моментом є незалежна перевірка ідентифікаційних даних користувачів кожною системою окремо (або організацією, якій власник системи довірив ідентифікацію своїх користувачів). Дуже часто ці методи не сильно відрізняються один від одного: користувачі передаються майже однаковий набір персональних даних, підтверджують реєстрацію за допомогою вже традиційних методів (електронна пошта, телефон тощо), внаслідок чого отримуються "локальний" ідентифікатор, що діє тільки в межах обраної системи [2, 23-25].

Основним недоліком такого підходу є те, що користувачеві знову й знову необхідно проходити однаковий процес ідентифікації і реєстрації, навіть якщо надається ідентичний набір даних. Окрім ресурсозатратності цих процесів, це тягне за собою те, що отримані в різних системах ідентифікатори ніяким чином не пов'язані між собою: інформація про ідентифікатори і підтвердження персональних даних (ПД) не синхронізується між постачальниками ідентифікаційних послуг, що в деяких випадках може призвести до використання різних ідентифікаторів для маніпулювання дозволами користувачів [25]. Більш того, при цьому, дані не зберігаються в єдиному уніфікованому форматі і не підписуються користувачем під час їх передачі, що тягне за собою складність пошуку зберігача певних даних і неможливість підтвердження їх цілісності та автентичності [23].

Глобальні сервіси, такі як Google [26], Facebook [27], Twitter [28], GitHub [29] та інші, дозволяють користувачам входити в інші сервіси через протокол OAuth [30], але цей протокол не забезпечує необхідної надійності даних криптографічними методами і використовує механізми сеансу для отримання доступу до даних користувача. Існуючі децентралізовані платіжні системи показали, що найкращою практикою є криптографічний підпис кожного запиту,

що надсилається до облікової системи, та підпис кожної відповіді, яку система повертає [31].

Глобальна система цифрової ідентифікації передбачає прив'язку всіх ПД користувача і його відкритого ключа (як варіант – набору ключів) до унікального глобального ідентифікатора. Метою побудування такої схеми є:

1) Вся інформація про підтвердження персональних даних зберігається в єдиній системі. Використання механізмів цифрового підпису та зв'язування наборів транзакцій між собою дозволить забезпечити автентичність конкретних підтверджень ПД з прив'язкою подій за хронологією[32].
2) Цілісність і автентичність прив'язаних до аккаунту даних перевіряється виключно криптографічними методами (контрольним кореневим геш-значенням дерева Меркла)[33].
3) Управління персональними даними повністю контролюється їх власником, всі інші учасники системи можуть тільки підтвердити набір, що визначений користувачем. Для отримання персональних даних щодо конкретного учасника системи, постачальник ідентифікаційних послуг (ПІП) повинен звернутися до нього безпосередньо і отримати або відразу необхідний набір даних або дозвіл на отримання цих даних від іншого постачальника.

Ідентифікаційні дані (ІД) – інформація про суб'єкт / об'єкт фізичний або цифровий, яка однозначно виділяє його серед інших. Ідентифікаційні дані є підмножиною персональних даних та використовуються ПІП у процесі ідентифікації користувача [34].

Identity (ідентифікаційний обліковий запис) – це сутність, яка зберігає інформацію про об'єкт ідентифікації і безпосередньо пов'язана з його ідентифікаційними даними [35].

Системи керування identity – сукупність взаємопов'язаних правил, методів, процесів, засобів і технологій, спрямованих на вирішення проблеми спільного використання таких identity незалежно від місця їх зберігання. Існують варіанти

при яких зберігачі identity взаємодіють безпосередньо за запитом клієнта, розкриваючи свої особистості, або діючи під псевдонімами. Такі взаємодії можуть припускати наявності посередників, якщо зберігачі identity не мають технічної сумісності [35].

Питання, що пов'язані із забезпеченням достатнього рівня довіри до ППП (таким постачальником ідентифікаційних послуг в тому числі може виступати і сам користувач) можуть вирішуватися за допомогою формування рівнів довіри. Якщо рівень довіри є достатнім, то взаємодія відбувається безпосередньо, в іншому випадку можуть залучатися додаткові учасники, що підтверджують (або підвищують) рівень довіри до конкретного постачальника. На сьогоднішній день існує ряд патентів, які надають подібні архітектури ідентифікації. Далі ми розглянемо деякі з них.

1.1 Аналіз патенту US10078758B1

Система ідентифікації, що представлені у цьому патенті, передбачає крос індексацію ідентифікаційних даних в різних локальних базах даних. При цьому не передбачається формування єдиної бази даних а лише отримання інформації з різних джерел і привласнення крос-індексів. Модель довіри до використовуваних локальних баз будується на основі ризик-орієнтованого імовірнісного підходу шляхом формування таблиць довіри. Всі процеси порівняння відбуваються на централізованому сервері [36].

Дуже цікавою особливістю цієї системи є саме зберігання ідентифікаційних даних у різних базах даних, до яких можуть звернутися користувачі, які ці дані потребують. Тобто модель дуже схожа на функціонування мережі довіри – користувачі незалежно можуть визначити якому провайдеру ідентифікації довіряти. Але, так як усі ці дані фактично індексуються через централізовану сторону – то така система не зможе у повній мірі забезпечити відмовостійкість, та також має бути присутня довіра до

власника сервера індексації (він може впливати на прийняття різними користувачами різної інформації стосовно баз даних) [36].

1.2 Патент US10237259B2

Система, що запропонована у цьому патенті, побудована за принципом "брокера" між постачальниками послуг (services providers – SP) і кінцевими користувачами при присутності провайдерів ідентифікаційних послуг (рис. 1.1). В якості "брокера" використовується додаток користувача, без якого функціонування системи неможливо [37].

На розподілену основу покладені реєстри з ідентифікаційною інформацією про користувача. Такі реєстри можуть бути відкритими чи закритими (належати провайдерам ідентифікаційних послуг), зберігати інформацію в відкритому вигляді або її геш-значення. Відкриті реєстри можуть містити в собі посилання на приватні та інші. Політики зберігання інформації в реєстрах залежать від типу інформації, що зберігається [37].

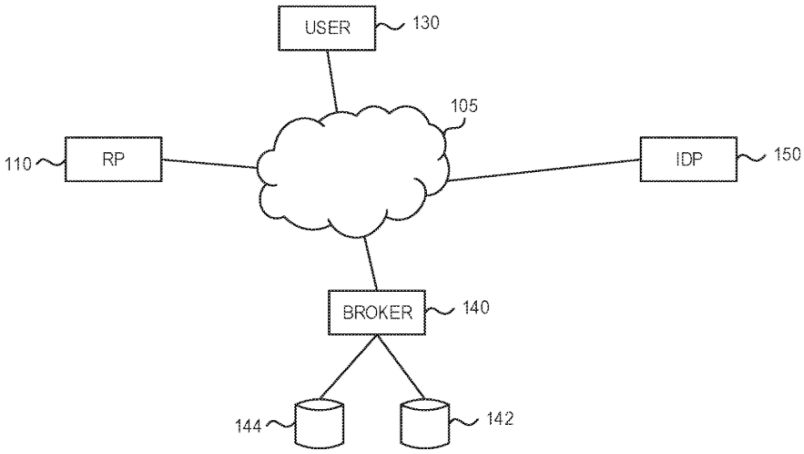

Рисунок 1.1 – Архітектура системи, запропонована в патенті US10237259B2

При цьому користувач може запросити інформацію з двох (або більше) реєстрів, в яких він значиться і звернутися до іншої сторони с "зібраною інформацією". Кожен користувач може формувати набір даних, який буде

зберігатися на зовнішньому ресурсі для відновлення доступу до свого довіреного гаманця без повторної реєстрації.

Як і в попередньому випадку, метою даного протоколу є розмежування зберігання даних про користувача та індексування за допомогою єдиного сервісу, але при цьому визначається необхідність використання довіреного додатку.

1.3 Протокол Open ID Connect

Специфікація OAuth 2.0 визначає протокол делегування, який надає клієнтам "безпечний делегований доступ" до ресурсів сервера від імені власника ресурсу (користувача). В основному, він визначає процес, щоб користувачі дозволяли сторонній стороні отримати доступ до своїх ресурсів сервера, без необхідності надання своїх облікових даних. Протокол призначений для роботи з HTTP і дозволяє серверу авторизації призначити маркери доступу стороннім клієнтам за погодженням спеціального власника ресурсу (рис. 1.2). Потім зворотний канал клієнта (сервер додатків) використовує маркер доступу для доступу до захищених ресурсів, розміщених сервером ресурсів. У більшості випадків сервер ресурсів і сервер авторизації однакові [38].

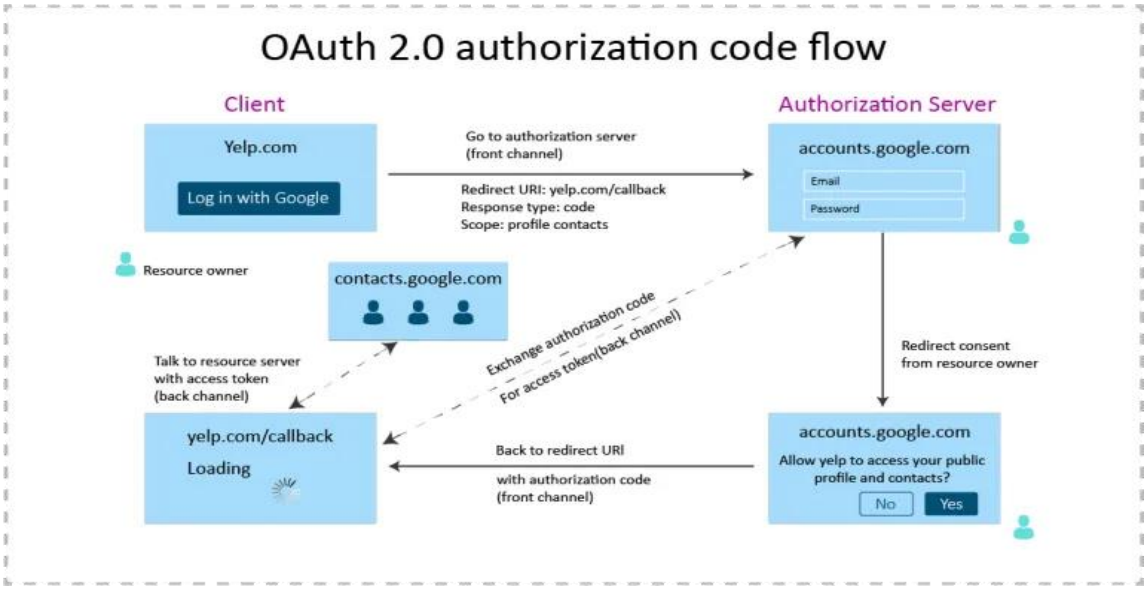

Рисунок 1.2 – Механізм функціонування протоколу OAuth

OAuth 2.0 використовується в широкому спектрі програм, будь то веб-або мобільних додатках. Він також використовується для надання механізмів аутентифікації користувачів [30]. Для цього використовується протокол верхнього рівня – OpenID Connect.

OpenID Connect – протокол автентифікації, розроблений поверх протоколу OAuth 2.0. Він дозволяє клієнтам підтверджувати особу та отримувати основну інформацію профілю щодо кінцевого користувача на основі автентифікації, виконаної за допомогою сервера авторизації. Розробники додатків та веб-розробників використовують його для автентифікації користувачів, не беручи на себе відповідальність за зберігання та керування паролями. Також цей протокол використовується для забезпечення функціонування методу єдиного входу (SSO) [39].

OpenID Connect робить можливою стандартну автентифікацію шляхом додавання декількох ключових компонентів на OAuth 2.0 (базовий шар). Ідентифікатор маркера OpenID Connect – це підписаний веб-маркер JSON (JWT). Він містить набір інформації про сеанс аутентифікації, включаючи ідентифікатори для кінцевого користувача, особу постачальника, який видав маркер, і клієнта, для якого цей маркер створений [30]. Клієнт аналізує вміст маркера ідентифікатора та отримує ідентифікаційні дані користувача. Після цього вже формується маркер доступу до ресурсу. Сервер авторизації підписує маркер ідентифікатора особистим ключем, який допомагає запобігти атакам підміни особистості.

Хоча на сьогоднішній день даний підхід багато де використовується, він має низку очевидних недоліків. По перше недоліком є необхідність повної довіри до постачальника даних (сервісу, що зберігає дані користувачів) від клієнту. Другий недолік полягає в тому, що провайдери ідентифікаційних послуг все ще не взаємодіють один з одним, це призводить до того, що бази даних провайдерів не синхронізовані і тому виникає неоднорідність при отриманні доступу до

ідентифікаційних даних користувачів. І третім недоліком цього протоколу є використання механізму сесій, які мають велику кількість вразливостей [40].

1.4 Аналіз патенту US20110010762A1

У даному патенті запропоновано поліпшення методу OpenID Authentication в комбінації з використанням федерації ідентичності (IF – identity federation) [30]. Схема відрізняється введенням додаткової ролі – сервісу валідації, який отримує ідентифікаційні дані і необхідний рівень, з яким їх потрібно підтвердити, проводить перевірку їх валідності, а також перевірку рівня надійності провайдера, який ці дані надав (рис. 1.3) [41].

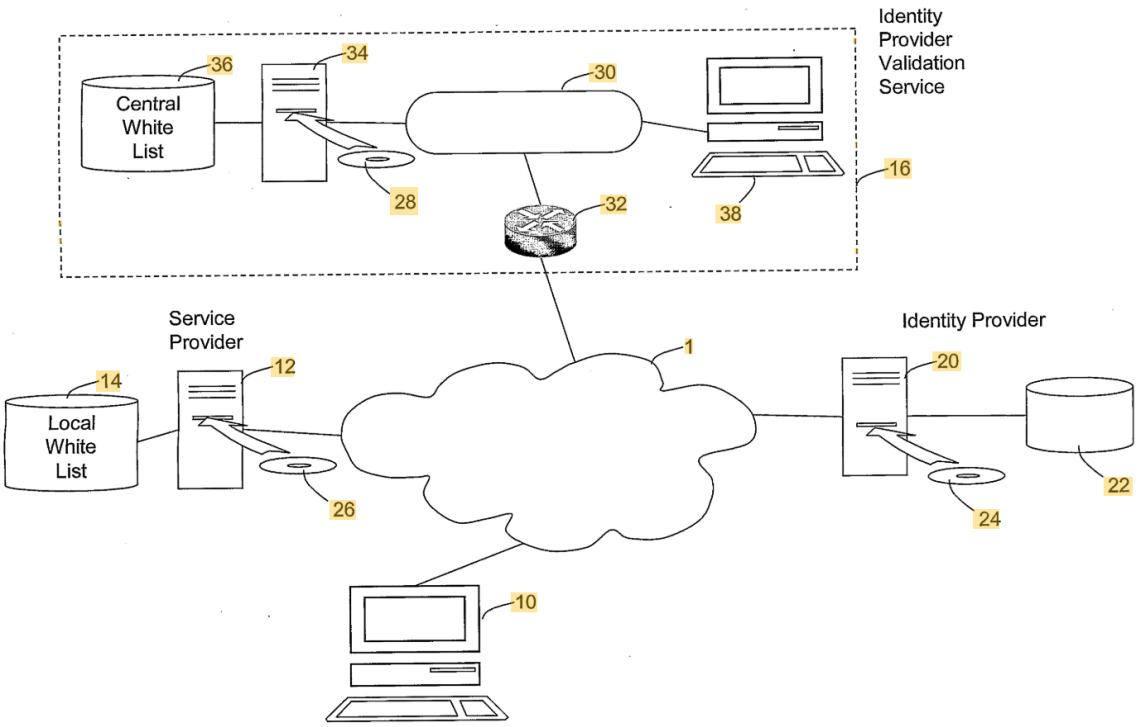

Рисунок 1.3 – Архітектура системи запропонована в патенті US20110010762A1

Якщо рівень надійності зазначеного провайдера недостатній, сервіс валідації може запросити перевірку наданих ідентифікаційних даних у інших провайдерів для досягнення заданого рівня надійності ідентифікаційних даних.

На основі накопиченої інформації від запитів формується список надійності провайдерів ідентифікації [41].

У цьому підході дуже цікавою особливістю є формування рівнів довіри відносно окремих провайдерів ідентифікації. Таким чином виконується відбір множини надійних провайдерів, які можуть повноцінно виконувати функції ідентифікації, що в свою чергу дуже сильно спростить процес ідентифікації кінцевих користувачів. Але знову, такий підхід потребує довіри до сторони сервісу валідації, що в значній мірі підвищує ризик її змови з окремим ПІП [42].

1.5 Патент WO2009133419A1

Цей протокол передбачає об'єднання учасників в групи, які формуються відповідно бажанням самих користувачів. У групі виділяється лідер – в більшості випадків він і виступає організатором групи. Лідер має можливість встановлювати політики безпеки і необхідні рівні підтвердження всередині окремої групи (рис. 1.4). При приєднанні нового учасника до групи, йому присвоюється відповідний маркер, що містить інформацію як про учасника, так і про групу, в якій він перебуває. Процеси авторизації групи проходять згідно з протоколом OAuth [30].

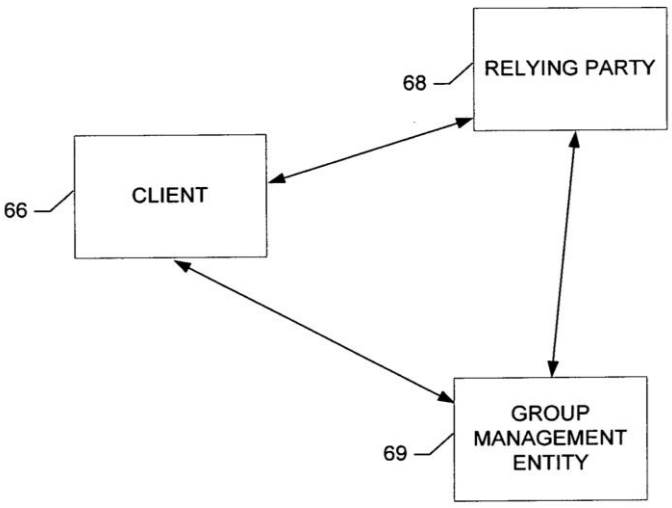

Рисунок 1.4 – Зв’язок клієнту, довіреної сторони та групи

Взаємна аутентифікація проводиться на підставі інформації з маркеру за участю довіреної сторони (чи декількох сторін в групі). Доказ надійності конкретного учасника ґрунтується на надійності групи, в якій він перебуває [43].

Таким чином клієнт, що потребує даних окремого користувача може збільшити рівень об’єктивності довіри отриманій інформації, так як вона фактично надана не єдиною довіреною стороною, а групою з незалежними учасниками. Але при цьому патент пропонує використання сесій для отримання доступу до даних користувачів, що супроводжується великою кількістю ризиків (перехоплення сесії та ін.) [40].

1.6 Аналіз патенту US7512649B2

Передбачається, що ідентичність кожного користувача встановлюється через її атрибути та репутацію. Система пропонує підхід для визначення репутації користувача за допомогою розподіленої мережі (P2P). Забезпечується можливість запросити репутацію щодо користувача від інших учасників мережі (до, під час або після конкретної дії цього користувача) [44].

Для підвищення рівня довіри до наданої на запит інформації клієнт може запитати інформацію щодо репутації окремих "рецензентів" - постачальників ідентифікаційних послуг. Репутація учасників даної транзакції може бути оновлена і поширена в мережу (при цьому немає вимоги, що вона повинна досягти все вузлів) при відправці окремої транзакції. Учасники транзакції можуть частково або повністю розкрити список інформації щодо, якому можна довіряти (блок такої інформації може бути розділений на складові частини, які при бажанні можна зібрати для отримання повної інформації).

У системі використовується платформа, що складається з трьох рівнів: рівень ядра, рівень сервісів, рівень додатків. Обов’язковим для кожного користувача є тільки нижній рівень, інші рівні можуть бути розгорнуті не на всіх вузлах (підтримуватися не усіма користувачами мережі) [44].

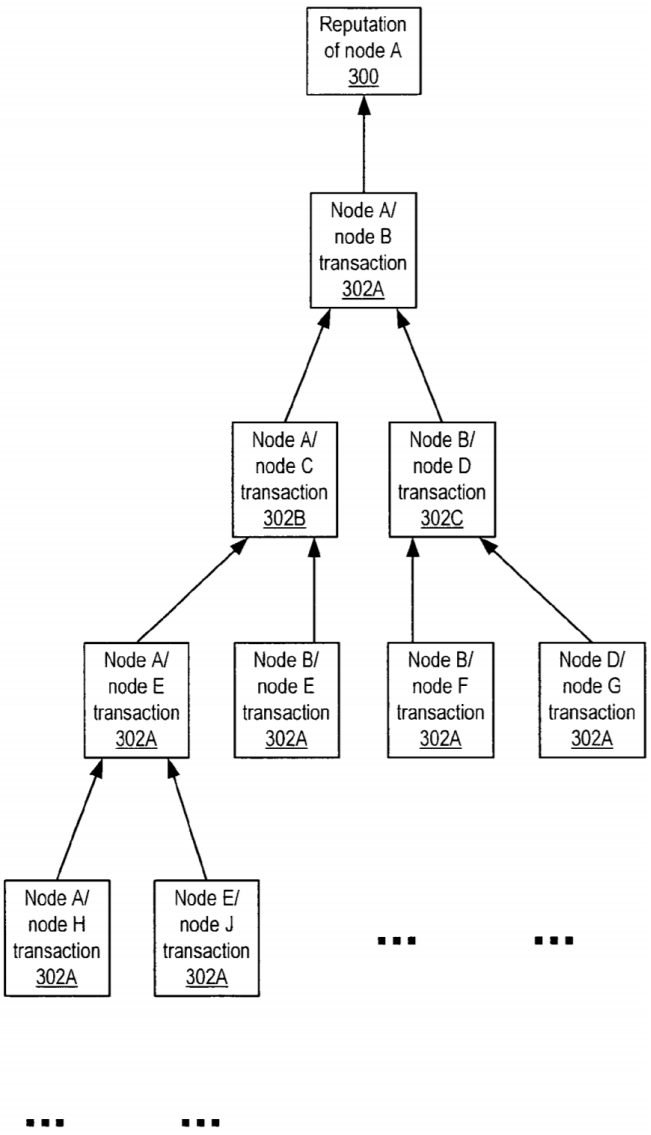

Рисунок 1.5 – Зв’язок репутації користувача з транзакціями учасників системи

Учасники мережі можуть бути об'єднані в групи (зазвичай учасниками групи стають користувачі чи провайдери ідентифікації з подібними характеристиками, наприклад, надають схожі сервіси); групи можуть перетинатися між собою, певні учасники групи можуть служити точками входу / виходу каналів, організованих між групами. Учасники в мережі можуть мати унікальний ID (його присвоєння передбачається за допомогою централізованого сервісу, наприклад: DNS, LDAP). Також у системі пропонується використання механізму "advertisement", який складається в публікації інформації про себе для пошуку схожих учасників [44].

Забезпечення базових послуг безпеки інформації всередині мережі може гарантуватися відповідними механізмами (наприклад, SSL, IPSec, PKI, PAM тощо) [45-46]. Така система є більш близька для застосування у реальному світі, так як вона може забезпечити можливість отримання даних про необхідну ідентичність, спираючись на голоси великої кількості учасників мережі. Але при цьому все одно не може забезпечуватися однорідність усієї системи ідентифікації, тобто ми не можемо одночасно повідомити усім вузлам про створення нової ідентичності чи компрометацію окремого ключа користувача тощо.

1.7 Аналіз патенту US+2019182035+(A1)

У цьому патенті запропоновано метод повторного використання ідентифікаційних активів користувача іншими провайдерами ідентичності. При цьому під ідентифікаційними активами розуміються дані, отримані в процесі ідентифікації користувача у будь-якого сервіс-провайдера, а не сам кінцевий ідентифікатор, виданий користувачеві (наприклад, клієнт банку при ідентифікації заповнював відповідну анкету – ідентифікаційним активом буде сукупність його відповідей на питання анкети) [47].

Взаємодія починається з клієнтського запиту до сервіс-провайдеру B, в якому клієнт вказує псевдонім сервіс-провайдера A, у якого можна отримати ідентифікаційний актив, сформований раніше. Сервіс-провайдер B публікує в мережу блокчейн транзакцію, в якій вказує псевдонім клієнта і сервіс-провайдера A (сервіс-провайдер B не знає справжньої особистості сервіс-провайдера A). У відповідь на транзакцію B, сервіс-провайдер A (упізнавши себе за своїм псевдонімом) надає ідентифікаційний актив клієнта (теж у вигляді транзакції, яка зберігається у блокчейні).

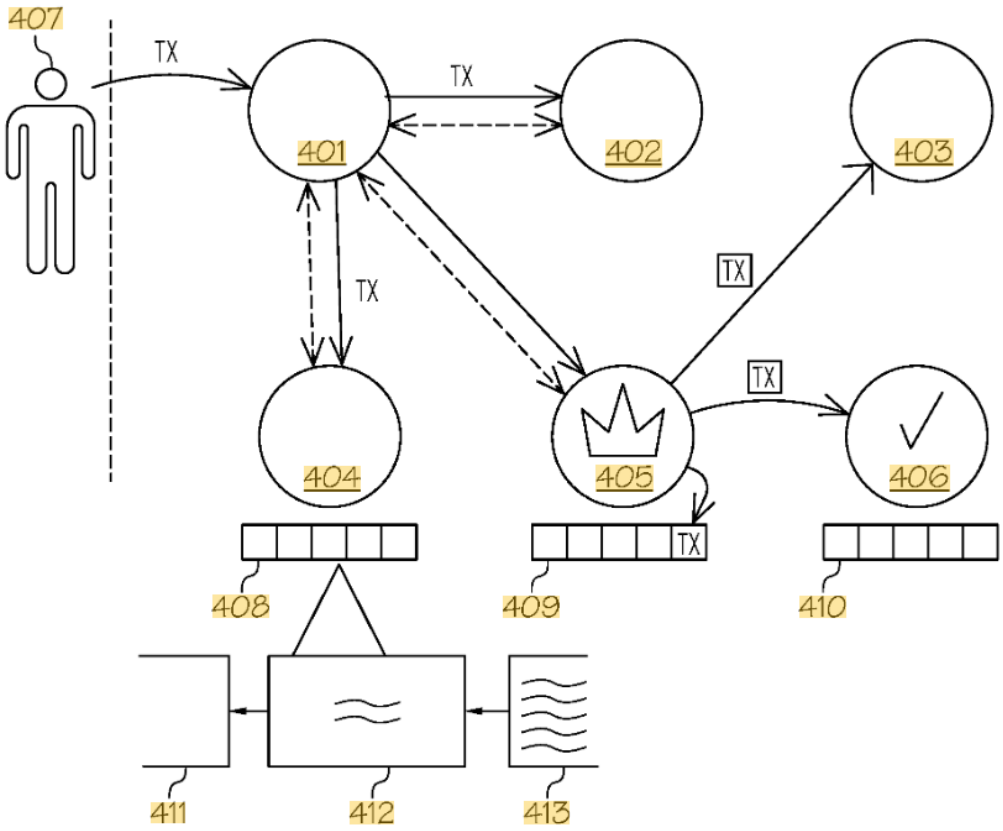

Рисунок 1.6 – Архітектура системи запропонована патентом US+2019182035+(A1)

Для провайдерів ідентифікаційних послуг передбачається монетарна мотивація з жорсткими умовами штрафів і винагород. Однак, додатково запропоновано удосконалений протокол анонімного обміну активами: сервіс-провайдер A зашифровує актив за допомогою ключа (K *), відомого сервіс-провайдеру B (метод передачі ключа залишається за рамками протоколу), і запитує у клієнта дозвіл на передачу його активу (теж за рамками протоколу). Після отримання підтвердження A генерує транзакцію, яку відправляє в блокчейн [47].

Даний протокол дуже гарно визначає спосіб передачі даних між провайдерами ідентифікаційних послуг між собою, але не визначає процеси ідентифікації та присвоєння сертифікатів відкритих ключів користувачам. До того ж виникають багато питань стосовно довіри ключам провайдерів (від

користувачів та інших провайдерів). Тобто така система не є самодостатньою і може функціонувати тільки поверх існуючої інфраструктури відкритих ключів та системи ідентифікації.

1.8 Підхід web-of-trust

Web-of-trust є альтернативою ієрархічної моделі ІВК. Концепція вперше була застосована в 1991 році в протоколі PGP для аутентифікації відкритих ключів користувачів [4-6]. Також підхід web-of-trust можна використовувати для механізмів ідентифікації в децентралізованому середовищі і системах рейтингу (вже існує безліч браузерних надбудов онлайн-рейтингу сайтів, що використовують механізм web-of-trust) [6].

В якійсь мірі в основі надійної роботи web-of-trust моделі ІВК лежить соціальний консенсус: жоден користувач не буде довіряти сертифікату іншого користувача до тих пір, поки цей учасник не отримає підтвердження свого відкритого ключа хоча б від одного учасника, що знаходиться в групі першого користувача.

Відзначимо, що при використанні web-of-trust неможлива відмова функціонування всієї системи через компрометацію особистого ключа одного або декількох користувачів. Якщо web-of-trust використовується в повністю розподіленому середовищі, то вихід з ладу одного вузла може вплинути тільки на можливість взаємодії з цим конкретним вузлом і сильно не вплине на функціонування інших вузлів між собою.

Однак технологія web-of-trust має ряд певних обмежень, які сильно впливають на роботу системи.

- складність знаходження сертифікату відкритого ключа з високим рівнем підтвердження в ряді випадків;
- велика тривалість і складність перебудови мережі;

- знаходження необхідного сертифіката може зажадати більш тривалого проміжку часу;
- база даних сертифікатів вимагає деякого об'єму пам'яті на вузлах.

1.9 Висновки

У першому розділі роботи розглянуто існуючи, у доступних джерелах, архітектури систем децентралізованої ідентифікації. Серед переваг розглянутих рішень можливо виділити рішення, що спрямовані на покращення характеристик систем ідентифікації:

- об’єктивність інформації щодо ідентифікаторів;
- зменшення залежності від єдиної сторони.

До недоліків можливо віднести:

- розглянуті системи ідентифікації не є самодостатніми: механізм покращення конкретних характеристик негативно впливає на інші характеристики системи;
- жоден з підходів не передбачає можливості синхронізації даних користувачів між постачальниками ідентифікаційних послуг (на глобальному рівні стан системи не однорідний).

Впровадження технології блокчейн і криптографії в системи цифрової ідентифікації може забезпечити як мінімум дві основні переваги:

- контроль користувачами персональних даних, що значно зменшить ризики, пов'язані зі зберіганням даних на централізованих серверах;
- системи для цифрової ідентифікації можуть бути надійнішими, на відміну від традиційних. Наприклад, завдяки використанню цифрових підписів можна відносно легко перевірити джерело ідентифікації користувача [48-53].
- використання технології блокчейн системою ускладнює фальсифікацію інформації та ефективно захищає всі види даних від шахрайства.

2 АРХІТЕКТУРА ДЕЦЕНТРАЛІЗОВАНОЇ СИСТЕМИ ІДЕНТИФІКАЦІЇ ТА ІНФРАСТРУКТУРИ ВІДКРИТИХ КЛЮЧІВ

Ідентифікатор є унікальним значенням, яке дозволяє однозначно ідентифікувати користувача в межах облікової системи. До кожного ідентифікатора в межах окремої облікової системи прив'язується аккаунт з відповідними балансами та/чи дозволами [35].

В межах облікових систем можуть використовуватися такі моделі отримання унікального ідентифікатора для користувача: випадково згенероване значення; ідентифікатор, що заданий самим користувачем (логін); відкритий ключ користувача; геш-значення від email або номера телефону.

Перші три з перерахованих способів активно використовуються на сьогоднішній день, однак вони мають одне обмеження: в межах облікових систем в такому випадку не можуть проводиться жодних дій з аккаунтами до їх створення користувачем.

В четвертому випадку ми можемо використовувати в якості ідентифікатора хеш-значення від email або номера телефону клієнта. Перевага цього підходу полягає в тому, що окрема облікова система може створити в себе обліковий запис користувача і виконувати дії з ним ще до моменту реєстрації самого користувача в системі (при реєстрації користувач визначає параметри доступу до облікового запису).

Але в цьому випадку також існує одна істотна проблема: ідентифікатор строго прив'язаний до електронної пошти: якщо користувач змінить адресу електронної пошти, йому буде необхідно створювати новий аккаунт і відправляти всі активи на нього (в межах кожної з облікових систем). Більш того, невідомо чим будуть активно користуватися користувачі через 10 років (раптом це буде не пошта, а інший зручний метод ідентифікації та звернення, в результаті чого знову доведеться змінювати ідентифікатори і все що з ними пов'язано) [4].

2.1 Глобальний ідентифікатор та дані, що з ним пов’язані

Основна роль глобальної цифрової системи ідентифікації: зв'язок глобального ідентифікатора користувача з його відкритим ключем і персональними даними, які цьому користувачеві належать, за допомогою перевірки зв'язку з декількома незалежними провайдерами ідентифікаційних послуг [21].

Глобальний ідентифікатор є унікальною в межах системи ідентифікації сутністю, яка представляє конкретний суб'єкт і пов'язана з інформацією, що ідентифікує даний суб'єкт. До ідентифікатору прив'язуються відкритий ключ власника ідентифікатора, набір геш-значень від ідентифікаторів інших облікових систем, а також набір геш-значень від ПД, які прив'язані до власника аккаунта. Всі перераховані дані зв'язуються в одну структуру – аккаунт (рис. 2.1).

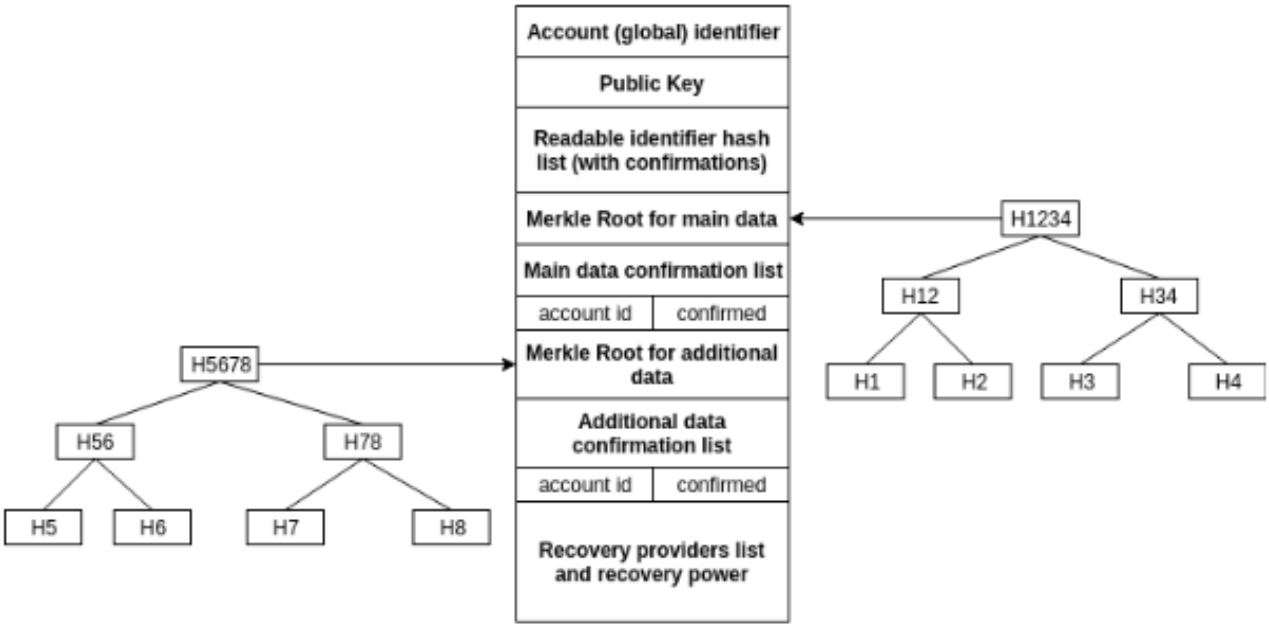

Рисунок 2.1 – Структура аккаунта в децентралізованій системі ідентифікації

«Account (global) identifier» – це незмінне і унікальне значення в системі ідентифікації, представлене у вигляді байтового рядка. Це значення формується, коли постачальник ідентифікаторів створює новий обліковий запис користувача.

Основна вимога – унікальність цього значення в системі (він може бути як згенерований, так і вибраний не випадковим чином).

«Public key» – значення відкритого ключа, особистим ключем якого підписуються всі транзакції, що ініціюються конкретним аккаунтом.

«Readable identifier list» містить список геш-значень ідентифікаторів у інших системах, таких як адреса електронної пошти, ідентифікатор facebook та ін. Також до кожного геш-значення прив'язується конкретний набір підтверджень від інших учасників мережі, що вони перевірили, що власник аккаунта володіє вказаною електронною поштою, facebook_id тощо.

«Merkle Root for main data» є кореневим значенням від основного набору персональних даних користувача. Передбачається, що значення, які входять до цього набору, не будуть часто оновлюватися і тому вони залишаться підтвердженими навіть при зміні інших полів аккаунта.

«Main data confirmation list» складається з набору записів, кожен з яких містить ідентифікатор identity провайдера та інформацію про підтверджений ним набор основних даних.

«Merkle Root for additional data» є кореневим значенням від додаткового набору персональних даних користувача. Передбачається що значення, які входять в цей набір будуть змінюватися, і потребувати підтверджень частіше, ніж основний набір.

«Additional data confirmation list» складається з набору записів, що складаються з ідентифікатора identity провайдера та інформації щодо підтвердженого ним набору додаткових даних.

«Recovery providers list and recovery power» визначає набір постачальників ідентифікаційних послуг (та їх кількість), необхідних для відновлення доступу до аккаунту і зміни відкритого ключа.

2.2 Структура транзакцій та блоків у системі

Глобальна система цифрової ідентифікації передбачає, що кожен учасник (який має обліковий запис) може відправити транзакцію до мережі. Транзакції зберігаються як упорядкований набір блоків (рис. 2.2), кожен з яких містить криптографічне геш-значення попереднього, що забезпечує можливість контролювати цілісність всієї історії подій, пов’язаних із глобальними ідентифікаторами.

Рисунок 2.2 – Структура та зв’язок блоків транзакцій

Транзакція, в свою чергу складається з набору заданих операцій. Всього існує чотири основні типи операцій: операція створення аккаунта, операції зміни/додавання даних, пов'язаних з конкретним глобальним ідентифікатором; операції підтвердження даних, пов'язаних з ідентифікатором; операція відновлення доступу до аккаунту (зміна публічного ключа і набору сторін для відновлення).

Операція створення аккаунта може бути ініційована будь-яким існуючим обліковим записом. В результаті виконання операції, в системі ідентифікації створюється новий аккаунт і обов'язково визначаються його ідентифікатор і відкритий ключ (додатково можна визначити поля, що містять геш-значення персональних даних та їх підтвердження).

До операцій зміни даних відносяться операції, метою яких є зміна / додавання основних і додаткових персональних даних користувача, а також зміна значень ідентифікаторів, прив'язаних до зовнішніх систем. Такі транзакції можуть бути підписані тільки власником облікового запису і перевірятися за допомогою зазначеного в аккаунті значення відкритого ключа. Фактично кожен

користувач особисто визначає набір даних, який хоче пов'язати з глобальним ідентифікатором, identity providers в свою чергу мають право тільки підтверджувати встановлені користувачем дані.

До операцій підтвердження даних відносяться операції, метою яких є підтвердження основних і додаткових персональних даних. Такі транзакції можуть бути відправлені identity providers (фактично будь-яким учасником мережі) по відношенню до конкретного аккаунту. Така транзакція містить в собі інформацію про ідентифікатор облікового запису, по відношенню якого виконується підтвердження, про значення Merkle Root, до якого відносяться перевірені їм денні, а також про набір самих даних, які були перевірені.

Операція відновлення доступу до аккаунту може бути підписана лише власниками ідентифікаторів, які були визначені користувачем в момент створення або оновлення аккаунта. Фактично користувач особисто визначає набір довірених сторін (і їх необхідну кількість), які можуть підтвердити його особистість і підписати транзакцію оновлення відкритого ключа аккаунта (рис. 2.3).

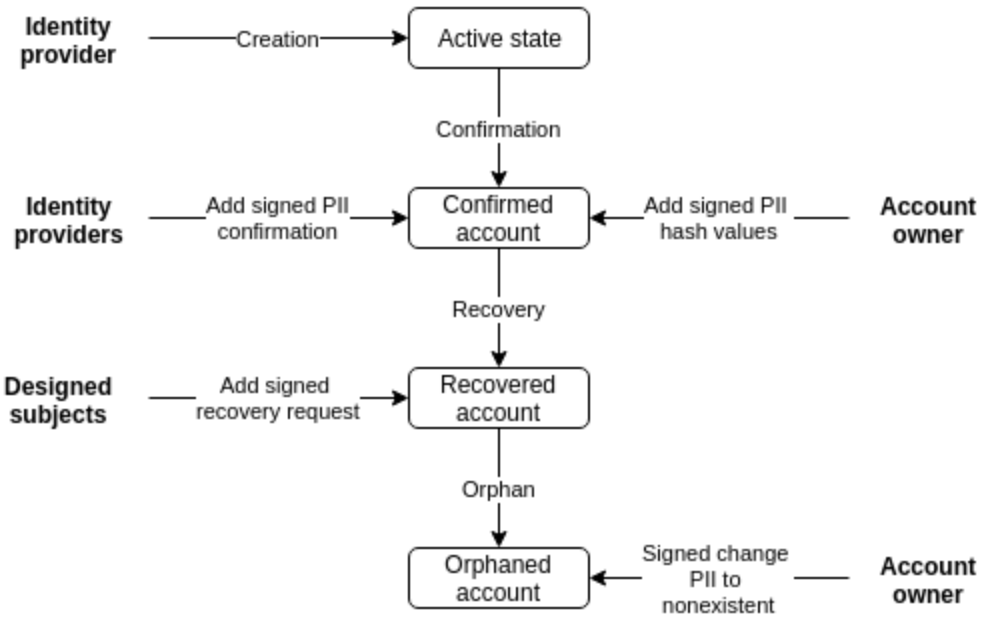

Рисунок 2.3 – Життєвий цикл аккаунту

2.2.1 Процес створення аккаунта

Для того, щоб створити обліковий запис у глобальній системі цифрової ідентифікації, користувачеві необхідно зв’язатися з постачальником послуг. Він може бути або окремим незалежним суб'єктом, або прив'язаним до певної системи обліку. Процес реєстрації може відрізнятися залежно від окремого постачальника (це може бути або особиста присутність користувача, або віддалена реєстрація), але варто визначити основні особливості цього процесу. У процесі створення облікового запису користувач надає постачальнику набір персональних даних, які повинні бути підтверджені, значення Merkle Root з усього набору даних, а також значення Merkle Branch для підтвердження того, що конкретний набір відноситься до значення кореня. Це дозволяє постачальнику перевіряти лише конкретні дані та погодитися, що вони включені до загального набору, і все це без надання всього набору персональних даних.

Також користувачеві необхідно створити пару ключів (особистий та відкритий ключі) та підтвердити право власності на вказану електронну адресу (або інший ідентифікатор, залежно від вказаних у обліковому записі). Процес створення облікового запису показаний на наступній схемі (рис. 2.4).

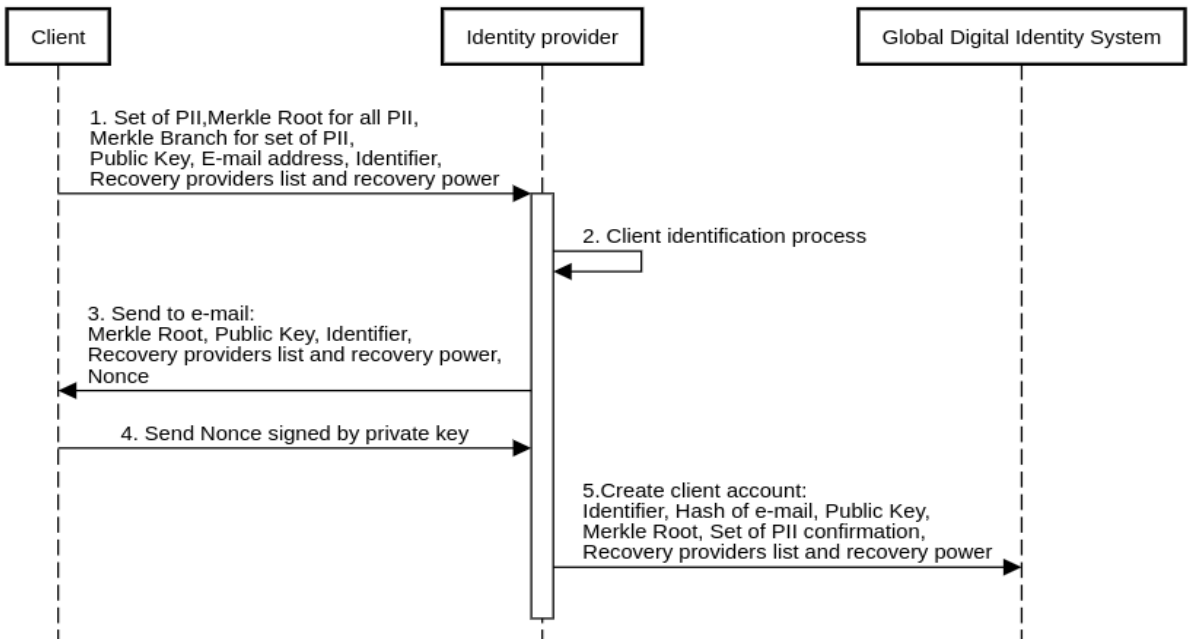

Рисунок 2.4 – Процес реєстрації користувача та створення аккаунта у системі

Спочатку користувач надсилає постачальнику ідентифікаційних послуг набір персональних даних, які повинні бути підтверджені, значення Merkle Root для всього набору персональних даних, Merkle Branch як доказ того, що набір персональних даних включений у кореневе геш-значення, відкритий ключ, електронну пошту та згенероване значення ідентифікатору. Користувач також визначає список аккаунтів, які можуть змінити значення відкритого ключа та відновити доступ до облікового запису (крок 1).

Далі постачальник ідентифікаційних послуг перевіряє надані персональні дані та перевіряє їх відношення до кореневого значення (крок 2).

Реєстратор надсилає клієнту підтвердження електронної пошти (не обов’язково потрібна електронна пошта, може бути використаний інший канал). Підтвердження містить Merkle Root (доказ того, що дані, отримані провайдером, не були змінені), отриманий відкритий ключ (також, що він не був змінений), ідентифікатор, список надійних постачальників та значення Nonce. Усі значення підписуються ключем провайдера; відповідно, атака на зміну повідомлення під час передачі виключається (крок 3).

Користувач отримує пошту, перевіряє, чи всі дані є дійсними, потім підписує значення Nonce своїм особистим ключем (як доказ того, що він володіє відкритим), а потім надсилає підписане значення постачальнику ідентифікаційних послуг (крок 4).

Реєстратор перевіряє підпис, після чого ініціює створення облікового запису: він заповнює необхідні поля, формує глобальний ідентифікатор і відправляє відповідну транзакцію (крок 5).

2.2.2 Процес підтвердження персональних даних

Після того як користувальницький аккаунт був створений, користувач визначає набір персональних даних, які будуть пов'язані з його аккаунтом. Отримані дані він збирає в структуру дерева Меркла і поміщає кореневе значення в аккаунт. Як ми зазначали раніше аккаунт містить два значення

Merkle Root: для набору основних даних і додаткових. Такий підхід дозволить залишити підтвердженими основні дані, навіть якщо додаткові будуть змінені.

Після того як користувач визначив персональні дані, йому необхідно отримати підтвердження цих даних від identity providers. При цьому важливо відзначити що користувач може не захотіти щоб провайдер отримав доступ до всього набору його персональних даних. Цю властивість можна забезпечити при використанні дерев Меркла. В цьому випадку в процесі підтвердження персональних даних, користувачеві необхідно передати не весь їх набір, щоб провайдер перевірив їх цілісність і автентичність, а тільки необхідні для перевірки дані і значення Merkle Branch для доказу їх входження в кореневе значення. (рис 2.5)

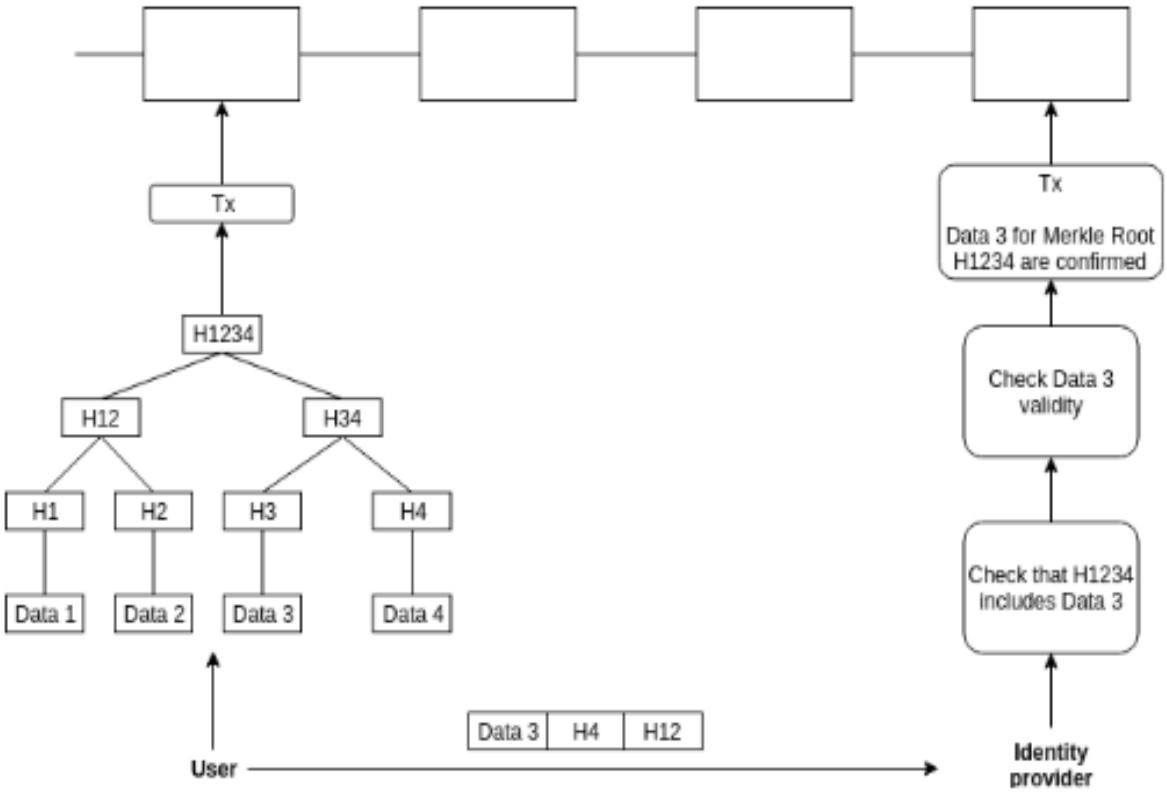

Рисунок 2.5 - Процес додавання та верифікації персональних даних

Коли identity provider отримує від користувача набір даних і Merkle Branch, він перевіряє що отримані дані відповідають вказаним в аккаунті кореневим геш-значенням, після чого виконує перевірку отриманих даних [32].

У разі якщо провайдер погоджується з отриманими даними, він формує транзакцію, в якій зазначено що він підтвердив конкретний набір персональних даних, які відносяться до конкретного ідентифікатора облікового запису і до конкретного значення Merkle Root.

У разі якщо користувач вирішить відновити частину даних для повного набору, в цьому випадку, значення Merkle Root повністю зміниться. Як результат - раніше відправлена транзакція, яка підтверджує дані для конкретного Merkle Root стає недійсною. Саме тому структура аккаунта має на увазі розподіл даних на основні та додаткові: якщо користувач підтвердив основний набір даних (і не змінює його), то незалежно від того чи були оновлені додаткові дані, основні дані залишаються підтвердженими.

2.3 Прийняття рішень щодо довіри ідентифікаторам користувачів

Важливою особливістю такої системи є незалежне прийняття рішень щодо ідентифікатора споживачем інформації. У системі немає інформації, яка безпосередньо визначає дійсність конкретного ідентифікатора: є лише аккаунти та голоси, що підтверджують дані цих облікових записів. Таким чином, питання довіри повністю передається на сторону клієнта. У цьому випадку він може особисто визначити методику, яка буде описана при розрахунку рівня довіри.

Одним із методів визначення довіри є кількість підтверджень ідентифікаторів іншими учасниками мережі. Однак слід мати на увазі, що цей метод піддається атаці Сивіли - один з облікових записів може створити велику кількість інших облікових записів, які підтверджують особу одного з учасників системи.

Другий підхід - довіряти лише конкретному постачальнику (декільком постачальникам). Наприклад, будучи клієнтом банку, я довіряю його механізмам ідентифікації і можу налаштувати своє забезпечення таким чином, щоб довіряти лише тим аккаунтам, які були підтверджені згаданим банком.

Хоча ця схема є дещо централізованою (якщо кількість провайдерів, яким користувач довіряє, невелика), вона буде досить ефективною і очікувано, найбільш використовуваною через її простоту.

У третьому випадку, якщо об’єктивність результату дійсно важлива для члена мережі, він може побудувати складні алгоритми перевірки, які оцінюють як рівень довіри для провайдерів, які підтвердили конкретний обліковий запис, так і постачальників, які підтвердили аккаунти у визначеного постачальника (таким чином до великої кількості рівнів перевірки).

Важливо зазначити, що незалежно від того, яким чином споживач використовує результати ідентифікації, система надає можливість повністю налаштувати алгоритм перевірки, який лежить виключно на стороні клієнта.

2.4 Дозвіл на отримання персональних даних користувачів

Щоб задовольнити політиці GDPR (англ. General Data Protection Regulation – загальний регламент щодо захисту даних) та Закону України Про захист персональних даних від 01.06.2010 № 2297-VI [77] необхідно щоб персональні дані користувачів передавалася і оброблялася тільки з дозволу їх власників. У разі якщо персональні дані передаються безпосередньо користувачем провайдеру, все досить просто: при встановленні з'єднання, користувач дає дозвіл на обробку своїх персональних даних. Проте складнощі виникають у разі якщо реєстратор запитує дані не безпосередньо у користувача, а у сторони, яка раніше провела його ідентифікацію (наприклад користувач не зберігає при собі весь необхідний набір даних для реєстрації на сервісі, проте ці дані раніше були підтверджені іншим реєстратором і відповідно у нього зберігаються).

Так як провайдер, який зберігає ПД користувачів не може поширювати ці дані, попередньо не отримавши на це дозволу, то такий дозвіл і має бути надано користувачем. Такий дозвіл має містити дату і час його формування,

ідентифікатор облікового запису власника персональних даних, набір даних, які дозволені для передачі другій стороні, а також ідентифікатор сторони-одержувача. Відзначимо, що запит повинен бути підписаний особистим ключем, який прив'язаний до аккаунту користувача.

Якщо реєстратор отримує від користувача подібний запит, він перевіряє значення підпису запиту використовуючи відкритий ключ, який вказаний в аккаунті користувача. Якщо підпис валідний, то він дивиться, який набір персональних даних запитується, і якщо у нього дійсно зберігається необхідний набір персональних даних, то він відправляє їх і значення Merkle Branch до запитуючої сторони (попередньо їм необхідно встановити захищений канал). Сторона, яка проводить запит, проводить процедуру, аналогічну до описаної в 2.2.2: перевіряє входження отриманих даних в кореневе хеш-значення, після чого підтверджує персональні дані.

Мітка часу в запиті необхідна для забезпечення конфіденційності оновлених персональних даних. У разі якщо користувач оновив свій набір персональних даних і не хоче щоб сторона, яка раніше отримала дозвіл на доступ до старого набору, могла отримати оновлені дані по тому ж дозволу. Тому при перевірці дозволу, реєстратор порівнює час відправки транзакції з оновленими даними і час отримання дозволу, і якщо час отримання дозволу менше, то доступ до персональних даних користувачів не надається.

2.5 Механізм досягнення консенсусу

Використання для мотивації валідаторів нативної для мережі валюти і використання PoW (proof-of-work) / PoS (proof-of-stake) механізмів досягнення консенсусу є не найбільш оптимальним рішенням, так як якість ідентифікаційної інформації та надійність системи, як показала практика, буде повністю залежати від "вартості цієї валюти". Більш того повністю анонімне середовище не найкраще за умовами до побудови системи ідентифікації, де

рівень довіри проведеної ідентифікації буде залежати від обчислювальної потужності або обсягу стека реєстраторів, що точно не свідчить про їх компетентність в питаннях ідентифікації користувачів.

Не варто змінювати модель довірчих відносин спеціалізованим постачальникам ідентифікаційних послуг, а необхідно тільки розширити її шляхом об'єднання цих постачальників в єдину мережу, яка буде містити в собі інформацію про видані ідентифікатори і їх підтверджень. При цьому провідні identity providers виступатимуть публічними джерелами інформації про підтверджених даних, що ніяким чином не вплине на зміну відносин до них. Іншими словами, довіряючи ідентифікацію певної організації, ви також продовжуєте їй довіряти, однак вона вже виступає не як постачальник ідентифікаторів для певного локального поля, а як постачальник ідентифікаторів в глобальній системі, що автоматично розширює призначену для користувача сферу діяльності. Таким чином підтвердивши ідентифікатор у кількох найбільших постачальників, користувач позбавляється від додаткової ідентифікації в локальних системах, які цим постачальникам довіряють.

2.5.1 Можливості використання pBFT алгоритму

Одним з методів досягнення консенсусу, прийнятний для даної системи може являтися pBFT алгоритм. Practical Byzantine Fault Tolerance (pBFT) [54] алгоритм досягнення консенсусу призначений для функціонування в асинхронних децентралізованих системах. Всі вузли в системі пов'язані між собою, причому в кожен момент часу один з вузлів є лідером. Метою протоколу є досягнення згоди між усіма чесними вузлами, коли кількість failed вузлів не більше $(n - 1) / 3$, де n – це загальна кількість вузлів-валідаторів в системі. У контексті BFT під failed вузлами маються на увазі не тільки ті, які не діють або поводяться непередбачувано, але ще й ті, які навмисно вступили в загальний змову з метою порушення роботи чесних вузлів [54].

Деяким обмеженням протоколу practical BFT є велика кількість повідомлень, якими валідатори обмінюються між собою. Відзначимо, що максимально можлива і мінімальна кількість таких повідомлень, коли можливе досягнення консенсусу, дорівнюють відповідно:

$$Messages_{max} = 2 * (n^2 - n) \quad (2.1)$$

$$Messages_{min} = 2 * (n^2 - n - f * n + f) + 1$$

де n – кількість всіх валідаторів, f – максимальна кількість failed вузлів (табл. 5.1) [32].

Таблиця 5.1 Кількість консенсусних повідомлень

Кількість валідаторів	Мінімальна кількість повідомлень	Максимальна кількість повідомлень
5	33	40
10	127	180
20	533	760
50	3333	4900

Тож ми бачимо, що швидкість обміну повідомленнями експоненційно зменшується залежно від кількості валідаторів. А у випадку, в якому система не висуває ніяких обмежень до кількості валідаторів та їхніх зв’язків між собою, функціонування pBFT протоколу в якості досягнення консенсусу не є ефективним підходом.

2.5.2 FBA алгоритм досягнення консенсусу

Найбільш ефективною моделлю досягнення консенсусу для даної системи є FBA алгоритм, який фактично на своєму рівні є проекцією довірчих відносин між учасниками мережі. Таким чином найбільшим постачальникам послуг буде вигідний запуск вузлів-валідаторів і налаштування зв'язку між ними. Таким чином кожен з них зможе забезпечити себе актуальною інформацією щодо стану ідентифікаторів користувачів, а зв'язки між глобальними

постачальниками не дозволять одній / або кільком сторонам шахраювати: в цьому випадку є великий ризик опинитися поза основними кворумів системи, в результаті чого втратити довіру з боку кінцевих користувачів і використовують ідентифікатори систем.

У порівнянні з Practical Byzantine Fault Tolerance, іншим протоколом з BFT-сімейства, Federated Byzantine Agreement забезпечує явну перевагу в плані масштабованості, оскільки не вимагає передачі великої кількості повідомлень між вузлами (у випадку), що дозволяє побудувати мережу з великою кількістю учасників [55].

Поділяючи всіх валідаторів платформи на групи довіри (quorum slices) протокол дозволяє відносно швидко досягти загального консенсусу в мережі, якщо розмір кожної групи не великий. Як видно з рисунків 2.6-2.7, то за збільшенням розміру quorum slice, зменшується загальна пропускна здатність системи та збільшується час повного підтвердження блоку (рис. 2.6 -2.7).

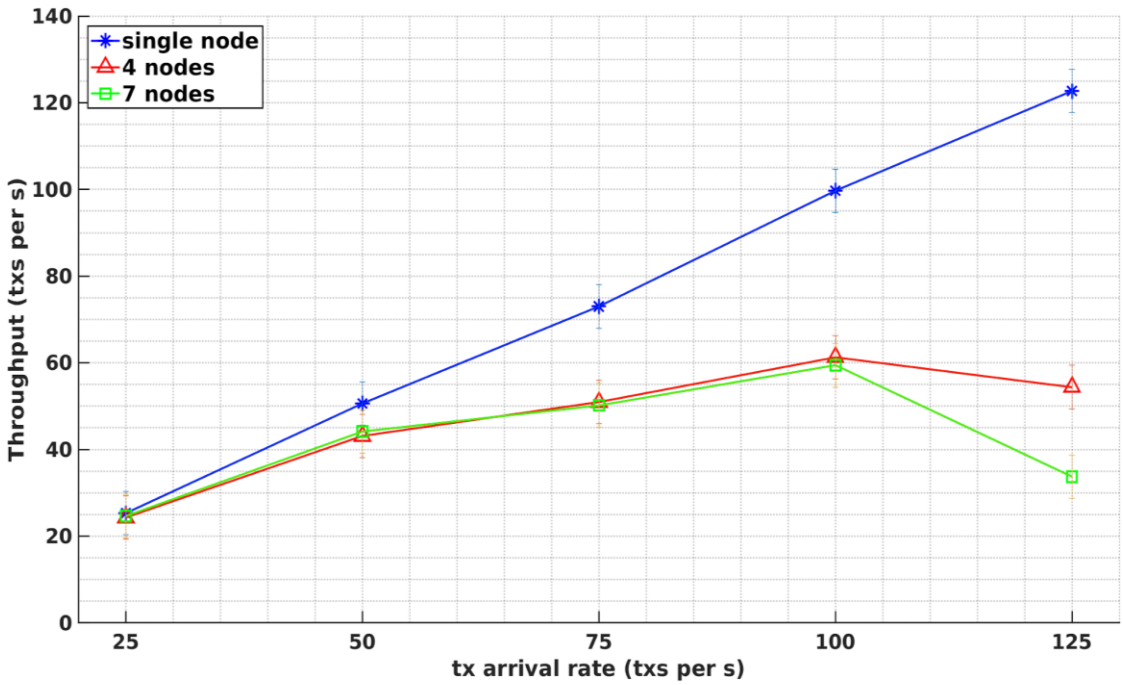

Рисунок 2.6 - Залежність пропускної здатності мережі від кількості валідаторів

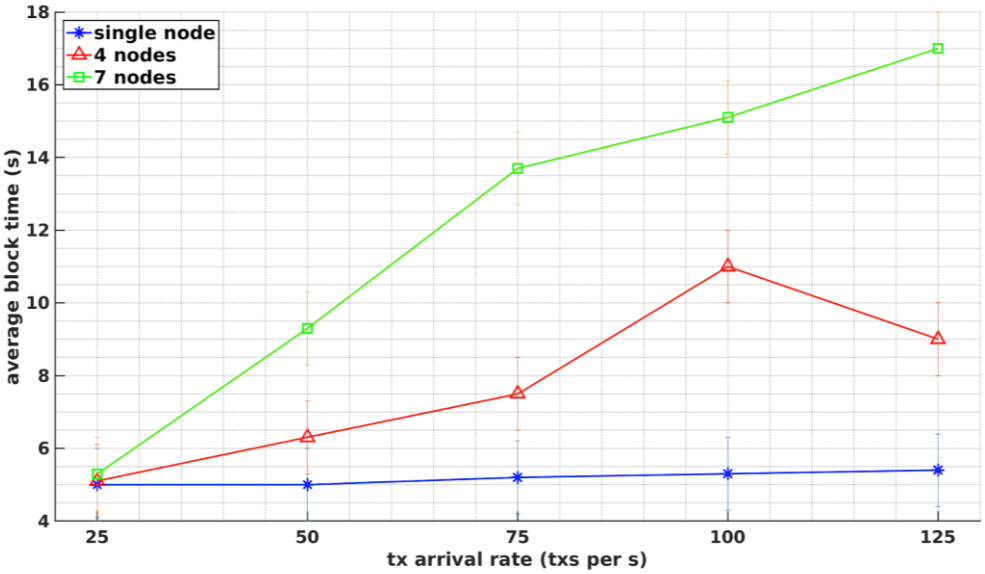

Рисунок 2.7 - Залежність часу закриття блоку від кількості вузлів

Однак, при порівнянні FBA з proof-of-work, останній має перевагу в плані анонімності учасників і забезпечує більший рівень децентралізації, оскільки кожен, хто має відповідне обладнання, може брати участь в процесі генерації блоків. Як було описано раніше в цьому розділі, Stellar, який є найбільш яскравим прикладом використання протоколу FBA на практиці, має потенційну схильність до централізованої структури, оскільки валідатори вважають за краще включати в свої quorum slices вузли, в чесності яких вони точно впевнені [32].

Ключовим питанням на даному етапі є захист від спам-атак: вони не можуть вплинути на механізм прийняття рішень щодо конкретного ідентифікатора, так як в разі довіри користувача тільки ряду провайдерів, їм можуть (і повинні) ігноруватися голоси інших учасників системи (цим способом ми захищаємося від атаки Сивіли); однак при цьому, так як будь-який користувач може додати транзакцію в мережу, і фактично кількість таких транзакцій не обмежена, то це може негативно відіб'ється на пропускній спроможності системи, тому повинен бути передбачений механізм захисту від цього з боку валідаторів (наприклад отримання винагород за додавання

транзакцій в блок - той же mempool, проте кожен реєстратор особисто домовляється з клієнтами щодо способу оплати; повторимо, що внутрішня валюта в такій системі має бути відсутня).

2.6 Висновки

Існуюча інфраструктура відкритих ключів була першим кроком у оцифровці взаємодій між користувачами Інтернету, що передбачав надання основних інформаційних послуг (конфіденційності, цілісності, автентичності). Вона показала, що ця взаємодія може бути здійснена за допомогою криптографічних методів на основі сертифікатів користувачів. Однак такі підходи все ще не є розповсюдженими, оскільки пов'язати дійсні (у різних системах обліку) ідентифікатори користувачів зі своїми сертифікатами досить важко. Інтеграція є складною, якщо мова йде про технічну та юридичну сумісність (включаючи використовувані алгоритми підпису), необхідність прямої довіри до послуг сертифікації та проблеми, пов'язані з синхронізацією інформації, пов'язаної зі створенням / оновленням / відкликанням сертифікатів. Також до недоліків можна додати єдину точку відмови та неефективність керування ключами у разі інфраструктура дуже велика (велика кількість користувачів та систем, що її використовують).

Впровадження схеми функціонування системи ідентифікації, описаної в документі, не передбачає обов'язкової відмови від існуючої інфраструктури X.509. Більше того, передбачається проста інтеграція існуючих сертифікатів шляхом додавання їх до структури рахунків. Ключова ідея такої системи полягає саме в передачі контролю даних у руки їх власників та уніфікації доступу до різних сервісів за допомогою криптографічних методів. Коли дані користувача прив’язані до певного ключа, право власності на яке підтверджує справжність суб’єкту, лише тоді система може бути впевнена в справжності запитувача.

Запропонована глобальна система ідентифікації та сертифікації може забезпечити надійну синхронізацію ідентифікаційних подій серед ППП за допомогою використання технології блокчейн. При цьому забезпечується повний контроль користувачами власних даних (зміна полів аккаунта може бути ініційована тільки їх власниками).

При цьому система дозволяє користувачам приймати незалежні рішення щодо прийняття окремих ідентифікаторів. Окрема ідентичність може бути підтверджена великою кількістю незалежних провайдерів, тобто рівень об’єктивності підтвердження даних користувача набагато вище, ніж у випадку використання централізованих провайдерів.

Крім того, така система здатна оперативно реагувати на компрометацію ключів користувачів, так як її стані є однорідними для всіх учасників і усі вузли в один момент часу можуть отримати інформацію щодо відклику окремого сертифікату. При цьому пропонується використання безпечного методу відновлення доступу до аккаунту, який полягає у зверненні до декількох провайдерів (яким довіряє користувач). Вірогідність змови всіх провайдерів (що підтримують різні методи автентифікації) є дуже низькою та дозволяє користувачу безпечно відновити доступ до свого аккаунту (хоча в деякому сенсі ускладнює проведення цієї процедури).

3 ДОСЛІДЖЕННЯ ФУНКЦІЙ ГЕШУВАННЯ ЩОДО ВИКОРИСТАННЯ ЇХ ДЛЯ ФОРМУВАННЯ ПОЛІВ У АККАУНТІ

Важливим механізмом для побудови описаної у другому розділі системі є вибір функції гешування для зв'язку персональних даних користувачів і додавання кореневого значення в структуру облікового запису учасника децентралізованої системи ідентифікації.

У цьому випадку функція гешування використовується для механізму контролю цілісності, а використання її в зв'язці з деревами Меркла дозволяють забезпечити контроль автентичності цих даних. В цьому випадку до використовуваних функцій гешування будуть висуватися наступні вимоги [56-58]:

- здатність відновлення в просторі і часі;
- стійкість до знаходження колізій;
- стійкість до знаходження прообразу;
- стійкість до атаки розширення повідомлення;
- відсутність побічних каналів витоку інформації.

До того ж, так як функція гешування буде використовуватися для відносно невеликих наборів даних (набори персональної інформації), які до того ж не мають властивості випадковості, то в цьому випадку функція гешування також повинна мати можливість працювати в режимі "розтягування" (для ускладнення відновлення даних) [56, 59-60].

3.1 Модель, що досліджується

Для проведення дослідження застосування геш-функцій для побудови дерев Меркла з персональних даних користувачів, опишемо основну структуру цього дерева.

Дерево Меркла є повним бінарним деревом, листям якого є геш-значення від довільних наборів даних. Листя і вузли дерева рівні між собою за розміром і залежать від геш-функції, що використовується. Кількість листів у дереві Меркла визначається як 2h, де h – висота дерева. Обчислення вузлів і кореня дерева Меркла здійснюється за такою формулою:

$$Hash_{i+1,\,j} = Hash(Hash_{i,\,2j} \,||\, Hash_{i,\,2j+1}) \qquad (3.1)$$

Для перевірки автентичності блоку даних, які входять до структури дерева Меркла, користувачем передається сам набір даних і набір значень Merkle Branch. Верифікатор перевіряє автентичність набору даних в такий спосіб:

$$MerkleRoot = Hash(Hash(Hash(Hash(Hash(Data) \,||\, MerkleBranch_0) \,||\, MerkleBranch_1) \,||\, \ldots) \,||\, MerkleBranch_n) \qquad (3.2)$$

В якості екземпляру цього дерева для дослідження, будемо використовувати дерево з висотою 3 і з кількістю листя, рівним 2^3. В якості вхідних даних для листя дерева будемо використовувати номер мобільного телефону користувача.

В якості досліджуваних геш-функцій, будемо використовувати:

- SHA-2 на довжині у 256, 384 та 512 бітів;
- SHA-3 на довжині у 256, 384 та 512 бітів;
- BLAKE2s на довжині 256 бітів,
- BLAKE2b на довжині у 256, 384 та 512 бітів;
- SCRYPT на довжині 512 бітів у декількох режимах в залежності від параметру складності (16384 - 1048576).

3.2 Швидкість гешування алгоритмів

Важливою характеристикою будь-якої геш-функції є швидкість обчислення геш-значення в залежності від розміру вхідних даних [56, 60-64]. У таблиці 3.1 вказується швидкість обчислення різних геш-функцій. Обчислення

проводилися на Intel(R) Pentium(R) CPU N3710 @ 1.60GHz з x86_64 архітектурою системи з використанням одного потоку. Обчислення проводились 100 раз для різних вхідних значений розміром: 32 B, 512 B, 1 KB, 100 KB, 1 MB. В таблицю заноситься середнє значення вимірів.

Таблиця 3.1 Швидкість обчислення геш-функцій в залежності від розміру вхідних даних

Геш-функція	32B, с	512B, с	1KB, с	100KB, с	1MB, с
SHA2_256	0.000009	0.000018	0.000026	0.001368	0.013656
SHA2_384	0.000015	0.000018	0.000027	0.000892	0.008993
SHA2_512	0.000016	0.000023	0.000028	0.000889	0.008921
SHA3_256	0.000022	0.000023	0.000028	0.000987	0.009710
SHA3_384	0.000019	0.000035	0.000032	0.001345	0.012439
SHA3_512	0.000020	0.000038	0.000045	0.001806	0.018115
BLAKE2s_256	0.000003	0.000008	0.000016	0.000819	0.007883
BLAKE2b_256	0.000005	0.000011	0.000016	0.000955	0.009705
BLAKE2b_384	0.000005	0.000009	0.000015	0.000990	0.009742
BLAKE2b_512	0.000003	0.000008	0.000019	0.000983	0.009680

Для функції SCRYPT ми використовували вхідні дані аналогічного розміру, але при цьому змінювали параметр складності функції. Отримані результати наведені у таблиці 3.2.

Таблиця 3.2 Швидкість обчислення функції SCRYPT

Парам. скл.	32B, с	512B, с	1KB, с	100KB, с	1MB, с
2^{14}	0.141307	0.143084	0.142789	0.144408	0.165761
2^{16}	0.556809	0.550226	0.549819	0.560404	0.584835
2^{18}	2.197675	2.191477	2.207537	2.232517	2.234709
2^{20}	8.746515	8.763436	8.727621	8.839558	8.777238

3.3 Колізійні характеристики геш-функцій, що досліджуються

Під стійкістю до колізій можна визначити обчислювальну складність знаходження двох повідомлень *Mi* та *Mj*, таких, що

$$H(Mi) = H(Mj)\ , \qquad (3.3)$$

де $H(Mi)$ це геш-функція, що використовується.

Для оцінки виникнення колізії потрібно визначити кількість випадкових повідомлень Mi, які треба подати на вхід геш-функції для того, щоб з ймовірністю *P(n,k)* відбувся хоча б один збіг, тобто колізія [56].

У нашому випадку є вибірка з k значень з рівномірним законом розподілу, причому k може приймати значення від 1 до $n = 2^m$ ($k \leq n, m -$ довжина значення). Також позначимо *R(n,k)* як значення ймовірності того, що в групі з k подій не відбудеться колізія, тобто співвідношення 3.3 не буде виконано жодного разу. Так як *P(n,k)* та *R(n,k)* є зворотніми величинами, то для них можна записати [56]:

$$P(n,k) = 1 - R(n,k)\ . \qquad (3.4)$$

Так як кожне з k значень може приймати n різних значень, то загальна кількість різних способів, де ми можемо вибрати k значень без повторень буде дорівнювати:

$$N = \frac{n!}{(n-k)!}. \quad (3.5)$$

При цьому загальна кількість можливих варіантів буде дорівнювати:

$$N_{\text{заг}} = n^k. \quad (3.6)$$

Для того, щоб обчислити ймовірність відсутності збігів, потрібно співвіднести кількість варіантів без збігів до загальної кількості можливих варіантів [56]:

$$R(n,k) = \frac{n!}{(n-k)!} / n^k, \quad (3.7)$$

а можливість виникнення збігів як:

$$P(n,k) = 1 - \frac{n!}{(n-k)!}/n^k = 1 - \left[\left(1-\frac{1}{n}\right)\cdot\left(1-\frac{2}{n}\right)\cdot \ldots \cdot \left(1-\frac{k-1}{n}\right)\right]. \quad (3.8)$$

Якщо пригадати, що за малих значень x ($x \leq 0.1$) виконується вираз:

$$(1-x) \approx e^{-x}, \quad (3.9)$$

то вираз (3.8) можна представити у наступному вигляді:

$$P(n,k) = 1 - \left[e^{-\frac{1}{n}} \cdot e^{-\frac{2}{n}} \cdot \ldots \cdot e^{-\frac{k-1}{n}}\right] = 1 - e^{-k(k-1)/2n}. \quad (3.10)$$

Тобто колізія може виникнути з ймовірністю $P(n,k) = 1 - e^{-k(k-1)/2n}$. Якщо ми виконаємо логарифмування та спрощення виразу (3.10), то отримаємо наступне [56]:

$$1 - P(n,k) = e^{-k(k-1)/2n}$$

$$\ln(1 - P(n,k)) = -k(k-1)/2n$$

$$\frac{k(k-1)}{2n} = -\ln(1 - P(n,k))$$

$$k(k-1) = -2n\ln(1 - P(n,k))$$

$$k^2 - k + 2n\ln\big(1 - P(n,k)\big) = 0 \quad (3.11)$$

У результаті ми отримали рівняння, у якому є залежність трьох величин: кількості подій, загальна кількість можливості подій та ймовірність, з якою може виникнути колізія. Якщо припустити, що k^2 є набагато більшою величиною, ніж k, то останнім можна знехтувати. В результаті ми отримаємо рівняння [56]:

$$k^2 = -2n\ln\left(1 - P(n,k)\right), \qquad (3.12)$$

з якого може бути розрахована залежність k від довільного значення $P(n,k)$:

$$k = \sqrt{2\ln\left(\frac{1}{1-P(n,k)}\right)\cdot n} = 1.44\sqrt{\ln\left(\frac{1}{1-P(n,k)}\right)\cdot n} \qquad (3.13)$$

В таблиці 3.3 наведена залежність ймовірності знаходження колізії у геш-функціях з довжиною виходу 256, 384 та 512 біт.

Таблиця 3.3 Колізійні характеристики геш-функцій

Ймовірність виникнення колізії $P(n,k)$	Довжина виходу геш-функції		
	256 бітів, кількість вибірки	384 біти, кількість вибірки	512 бітів, кількість вибірки
0,1	$1.59 * 10^{38}$	$2.93 * 10^{57}$	$5,4 * 10^{76}$
0,2	$2.31 * 10^{38}$	$4.27 * 10^{57}$	$7.8 * 10^{76}$
0,4	$3.50 * 10^{38}$	$6.46 * 10^{57}$	$1.19 * 10^{77}$
0,5	$4.07 * 10^{38}$	$7.52 * 10^{57}$	$1.39 * 10^{77}$
0,6	$4.69 * 10^{38}$	$8.65 * 10^{57}$	$1.59 * 10^{77}$
0,7	$5.37 * 10^{38}$	$9.92 * 10^{57}$	$1.83 * 10^{77}$
0,8	$6,21 * 10^{38}$	$1,15 * 10^{58}$	$2.11 * 10^{77}$
0,99	$1,5 * 10^{39}$	$1.93 * 10^{58}$	$3.58 * 10^{77}$

Графічне представлення результатів таблиці 3.3 знаходиться на рисунку 3.1.

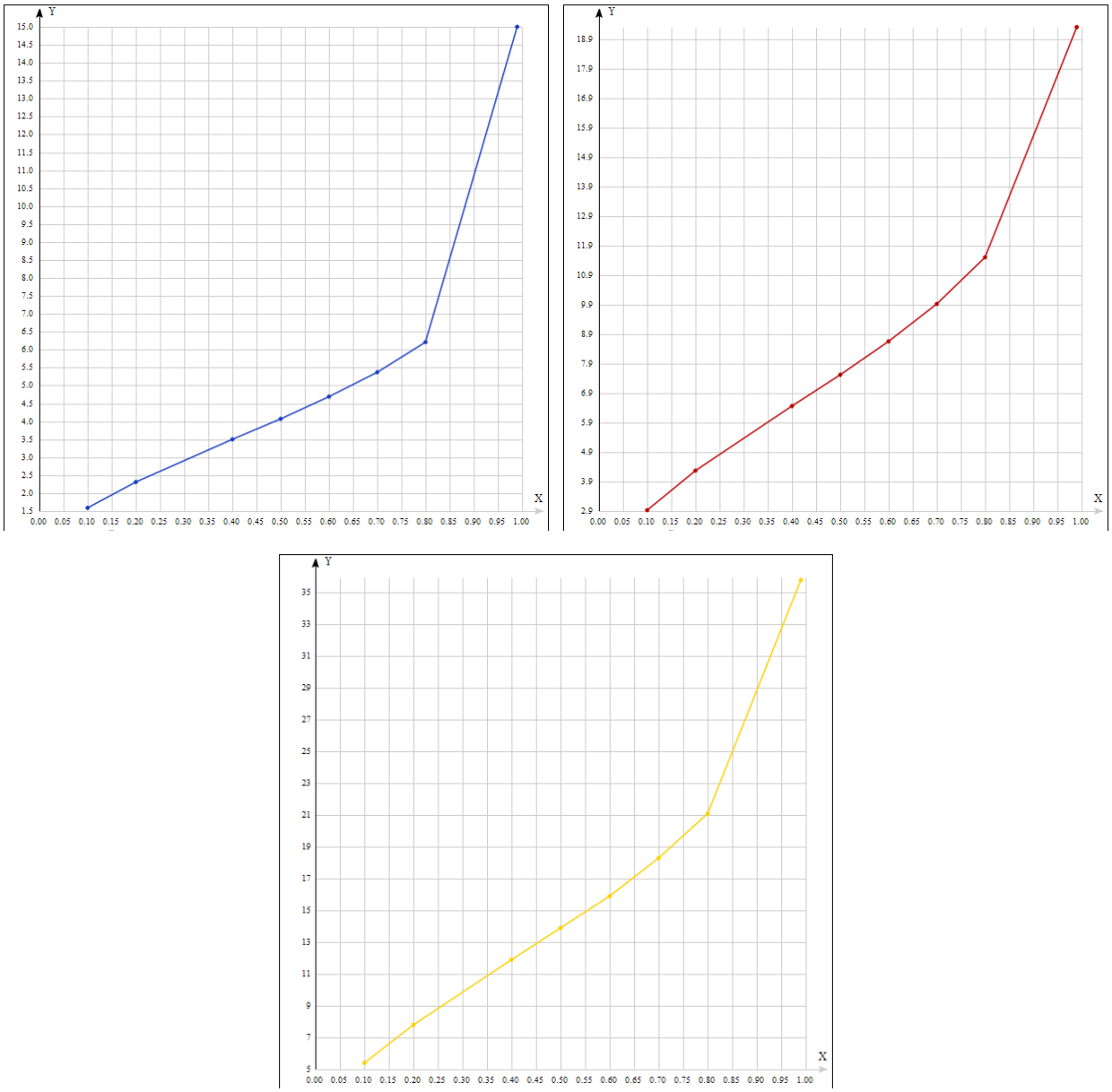

Рисунок 3.1 – Графіки залежності ймовірності знаходження колізії від розміру виходу геш-функції

3.4 Підбір номера телефону при відомому значенні Merkle Branch

Важливою особливістю при ідентифікації користувача у окремого провайдера ідентифікаційних послуг є надання їм необхідної інформації, яку він хоче підтвердити, а також набору значень Merkle Branch для того щоб підтвердити що окремі дані належать кореневому значенню [33]. Однак такий підхід може бути використаний зловмисником (тим же провайдером

ідентифікації), який може відкрити частину даних, використовуючи згаданий Merkle Branch (рис. 3.2).

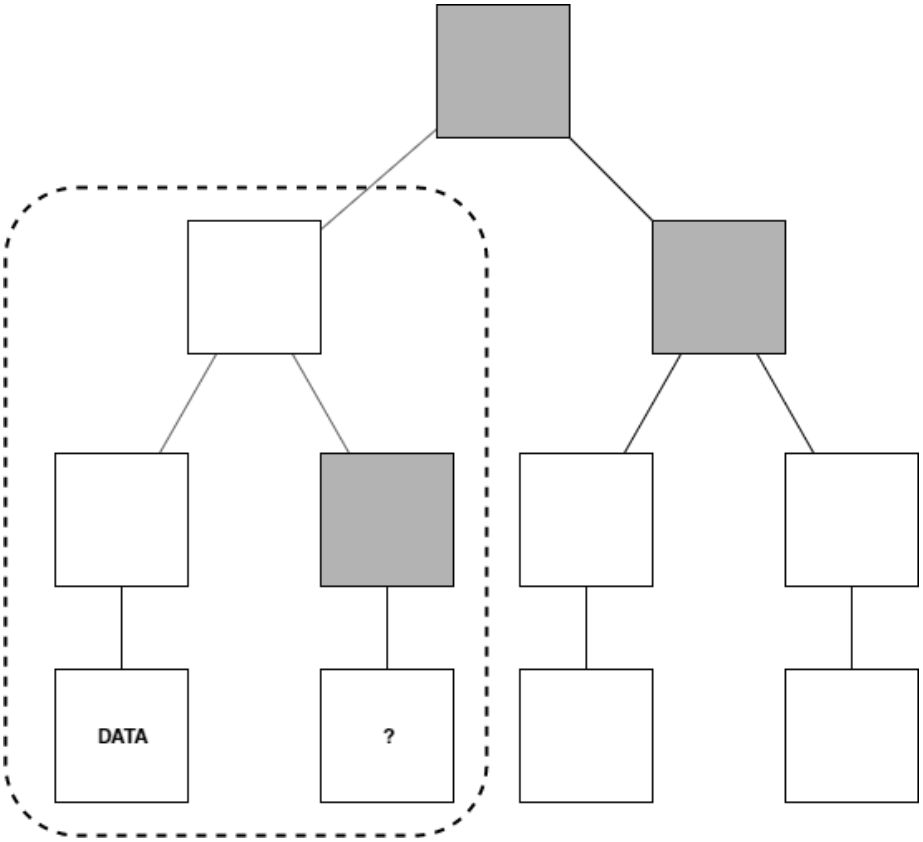

Рисунок 3.2 – Зв’язок даних з частиною Merkle Branch

Таким чином, зловмисник може перебирати набір даних, які позначені "?" на рисунку 3.1, і порівнювати їх з відповідним листом дерева, який у зловмисника є [32].

3.4.1 Атаки зі словником

Словникова атака – це метод розтину будь-якої інформації шляхом перебору можливих значень даної інформації. Словникові атаки часто класифікують як один з варіантів атаки методом «грубої сили», оскільки в ньому також виконується перебір варіантів даних. У порівнянні з методом «грубої сили» словникові атаки здійснюють перебір по істотно меншій безлічі можливих значень.

Один з варіантів реалізації атаки зі словником є атака, що базується на попередніх обчисленнях. Шляхом обчислення атакуючий отримує таблицю відповідностей, до якої входять дані та їх геш-значення. Для знаходження даних, відповідних відомому зловмисникові геш-значенню h, виконується пошук геш-значень в таблиці [56]. Основним недоліком даної атаки є досить великий обсяг пам'яті, необхідної для зберігання таблиці, розмір якого в бітах для n-бітного алгоритму гешування можна оцінити таким чином:

$$V = (n + k) * |P| , \qquad (3.14)$$

де n – довжина виходу геш-функції, k – довжина повідомлення, а $|P|$ – простір значень, що зберігаються.

Другим варіантом є використання ланцюжка геш-значень. Його покращення полягає в тому, що при використанні ланцюжків на зберігання таблиці паролів і відповідних їм геш-значень потрібно істотно менше пам'яті, ніж в разі класичної словникової атаки [78]. Принцип дії ланцюжків хеш-значень заснований на введенні додаткової функції *R()*, за допомогою якої будь-якого хеш-значенням h можна псевдовипадковим чином зіставити якійсь дані d з множиною всіх можливих значень.

$$d1 \rightarrow h(d1) \rightarrow d2 \rightarrow h(d2) \rightarrow \cdots \qquad (3.15)$$

Принциповим моментом є те, що для зменшення необхідної для зберігання пам'яті зберігається не вся таблиця, а лише перше і останнє значення (тобто $d1$ та $h(dn)$) для кожного рядка (де n – кількість елементів у рядку). Значення n є параметром таблиці ланцюжків, від якого залежать дві важливі характеристики таблиці:

- розмір пам'яті для зберігання таблиці ланцюжків обернено пропорційний значенню n при еквівалентному покритті безлічі паролів (без урахування колізій, ймовірність яких нелінійно зростає зі зростанням n);
- складність знаходження необхідного пароля в таблиці, яка лінійно зростає з ростом n.

3.4.2 Грубий перебір і реалізація атаки зі словником

Одним з набору даних, прив'язаних до аккаунту через дерево Меркла може бути номер мобільного телефону користувача. Тому для початку розглянемо складність підбору телефонного номера, який представляє з себе послідовність з 9 десяткових цифр (ми маємо на увазі українські номери). При грубому переборі, перебиралась вся множина номерів. Простір дійсних номерів будувалося з урахуванням можливих мобільних і міських номерів. Для атаки по словнику використовувався телефонний довідник з попереднім обчисленням та кількістю номерів: 29760300. У таблиці 3.4 представлено необхідну кількість часу для перебору описаних підмножин за допомогою різних функцій гешування.

Таблиця 3.4 – Характеристики підбору номеру телефону за значенням

Геш-фунція	Прост. усіх номерів, с	Прост. дійсн. номерів, с	Прост. моб. номерів, с	Прост. міських номерів, с	Атака за тел. словником, с
SHA256	4698	2866	893	1316	1578
SHA384	6417	3914	1219	1796	2117
SHA512	6398	3903	1216	1791	2111
SHA3_256	10102	6162	1919	2829	3334
SHA3_384	10131	6180	1925	2837	3343
SHA3_512	10261	6259	1950	2873	3386
BLAKE2s_256	1742	1063	331	488	575
BLAKE2b_256	2905	1772	552	813	959
BLAKE2b_384	3075	1876	584	861	1015
BLAKE2b_512	2981	1818	566	835	984

Виходячи з таблиці, ми бачимо, що максимальний час перебору (в разі атаки грубої сили і використання SHA3 на довжині 512 біт) - до 3-х годин. У разі перебору мобільних номерів, якщо дерево було побудовано за допомогою геш-функції BLAKE2s_256 - близько 6 хвилин.

В таблиці 3.5 представлено необхідну кількість часу для перебору номерів в разі використання функції SCRYPT і параметрів складності, які дорівнюють 2^{14}-2^{20} (значення в таблиці приблизні розраховані приблизно і були представлені виходячи з часу, необхідного для перебору 10^4 множини номерів).

Таблиця 3.5 Характеристики підбору номеру за використання SCRYPT

Парам. скл.	Прост. усіх номерів, год	Прост. дійсн. номерів, год	Прост. моб. номерів, год	Прост. місц. номерів, год	Атака за тел. довід., год
2^{14}	3976	2425	756	1113	1312
2^{16}	155051	94581	29460	43414	51167
2^{18}	613525	374250	116570	171787	202463
2^{20}	2784122	1698314	528983	779554	918760

Графічне представлення результатів таблиць 3.4 та 3.5 знаходиться на рисунку 3.3.

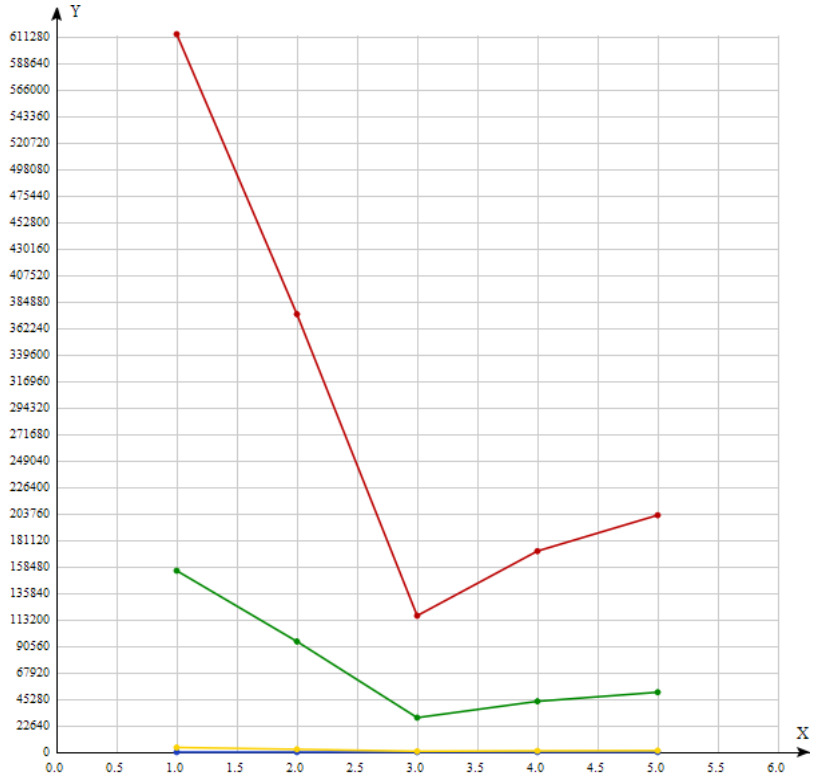

Рисунок 3.3 – Залежність швидкості від множини перебору

З першого погляду здається, що підібрати окрему адресу електронної пошти набагато складніше, так як простір всіх адрес електронної пошти набагато вище, ніж простір номерів мобільних телефонів. Так воно і є насправді, бо згідно зі стандартом RFC 821 [65] адреса пошти може складати 128 символів та «@». Проте за останніми підрахунками, на сьогодні існує тільки близько 5.6 мільярдів адрес електронної пошти [66]. Ця цифра близько у 2^9 більша, ніж простір усіх мобільних номерів України.

3.5 Висновки

Дослідження колізійних характеристик геш-функцій показало, що усі досліджувані функції можуть використовуватися для побудування дерев Меркла, що містять персональні дані користувачів (з точки зору знаходження колізії). Аналіз показав, що для виникнення колізії з ймовірністю хоча б 0,1,

потрібно, щоб простір таких даних дорівнював $1.59 * 10^{38}$, $2.93 * 10^{57}$, $5{,}4 * 10^{76}$ відповідно до довжини виходу геш-функції 256, 384 та 512 бітів (рис. 3.2).

Але при цьому необхідно приділити увагу використанню функцій, що висувають вимоги до пам’яті та часу обчислення. В цьому випадку реалізація атак грубої сили є набагато складнішим процесом, ніж при використанні швидкодіючих геш-функцій. Для порівняння, графік, що зображений на рисунку 3.3 показує відношення вимог до часу, що знадобиться для підбору номеру мобільного телефону за допомогою SHA3 на довжині 512 біт та часу, що знадобиться при використанні функції SCRYPT на тій самій довжині та з параметром складності $2^{14} – 2^{18}$.

ВИСНОВКИ

У сучасному світі, що розвивається в цифровому масштабі, традиційні підходи до управління доступом та управління ідентифікацією вже не ефективні через низку проблем:

- ідентифікатор працює лише в організації, яка його підтвердила;
- питання довіри до сертифікаційних органів усіх рівнів;
- необхідність перевірки всього ланцюжка сертифікатів користувачем (до кореня);
- проблеми, пов’язані з синхронізацією серверів протоколу статусу Інтернет-сертифіката (OCSP);
- складність та вартість експлуатації та розробки інтегрованих систем для постачальників ідентифікаційних послуг;
- складність інтеграції існуючих систем з новою хмарною ідентичністю або управлінням доступом як послуга.

Більшість постачальників ідентифікаційних послуг розширюють свої моделі розгортання за межами локальної мережі, використовуючи хмарні технології з можливістю підтримувати роботу системи у швидко мінливому середовищі. У цьому випадку блокчейн може вирішити проблему забезпечення додаткового рівня надійності та гнучкості для впровадження таких систем і уникнути дорогого обслуговування локальної системи.

Основні положення (такі як GDPR, Закон України Про захист персональних даних від 01.06.2010 № 2297-VI тощо) вимагають посилених заходів щодо регулювання та перевірки даних користувачів. Такі заходи застосовуються як до провайдерів ідентифікаційних послуг, так і до організацій, які використовують персональні дані користувачів.

Це означає, що власнику платформи потрібно рішення з управлінням ПД, що дозволяє забезпечити конфіденційність під час зберігання, обробки та

передачі даних. Блокчейн – це рішення, яке може забезпечити надійну систему ідентифікації в середовищі з множною незалежних сторін, дотримуючись політики безпеки щодо конфіденційності даних користувачів.

Існує повністю децентралізований підхід, заснований на концепції веб-довіри [67-71]. Суть підходу полягає в тому, що користувачі організовують систему однорангових вузлів, і кожен з них займається ідентифікацією та сертифікацією інших учасників мережі (тобто, кожен користувач вирішує, чи слід довіряти ідентифікатору іншого користувача). Цей метод організації має безперечні переваги, такі як неможливість впливати на систему та більш об'єктивну інформацію про те, чи відповідає ідентифікатор учаснику системи.

Але такий підхід також має ряд обмежень, що знижують ефективність цих систем:

- тривалий процес відновлення мережі довіри (у разі зміни ідентифікатора користувача, оновлення системи займе дуже багато часу, оскільки 1) більшість учасників системи мають оновити ідентифікатор 2) і тому, що потрібно переконати учасників що зміна ідентифікатора ініціює саме цільовий користувач, а не зловмисник;
- користувачі не готові надати всі свої дані, пов’язані з ідентифікатором у відкритому доступі.

Існує ряд пропозицій, заснованих на використанні стандарту X.509, але із застосуванням технології блокчейн. У цьому випадку сертифікати відкритого ключа додаються до блокчейн, який незалежно підтримується органами з сертифікації. Такий підхід полегшує процес синхронізації органів сертифікації та забезпечує незмінність історії діяльності органів сертифікації [72-75] та має низку переваг:

- швидкість та простота перевірки сертифікату. Загальнодоступна база даних впорядковує усі дії з сертифікатами. Будь яка сторона може перевірити ким конкретній сертифікат був виданий та перевірити його актуальність.

- немає потреби у сервісі який видає списки відкликаних сертифікатів що є відомою з перших років застосування проблемою традиційних ІВК. Інформація щодо відкликання сертифікатів також оновлюється у єдиному реєстрі;
- немає потреби відповідати на запити протоколу онлайн статусу сертифікатів, а також розкривати інформацію що вказує на подання суб'єктом сертифікату;
- ІВК на БЧ можна використовувати для резервування простих сертифікатів;
- ІВК на основі БЧ забезпечує гнучкий захист від атак «людина по середині».
- хоча центральний орган з сертифікації все ще залишається, при цьому вже підтверджені сертифікати можуть не бути відкликані, так як вони були підтверджені до компрометації ключів головного центру, і такий стан єдиний для всіх користувачів.

Архітектура, що запропонована в даній роботі має одну ключову перевагу: передача управління даними та прийняття рішень на сторону кінцевого користувача. Таким чином пропонується об'єднання підходів: використання БЧ для синхронізації подій між незалежними сторонами, так щоб кожна з них мала однаковий стан локальної бази даних; та прийняття кожною стороною рішень незалежно, орієнтуючись тільки на власний стан бази даних.

Торкаючись можливості побудування дерев Меркла для персональних даних користувачів, то цей механізм може використовуватись, але при цьому необхідно висувати вимоги щодо геш-функцій, що використовується. Можливість проведення словникових атак на криптосистеми обов'язково повинна враховуватися при розробці систем захисту.

Найбільш дієвим методом захисту від райдужних таблиць і інших видів словникових атак є доповнення (наприклад, шляхом конкатенації) значення, що гешується, будь-якої випадкової величиною (сіль), яка згодом зберігається разом з геш-значенням. У цьому випадку атакуючий повинен розглянути також множину значень цієї величини.

Ще один варіант протидії – багаторазовий прогін значення через функцію гешування, що багаторазово збільшує необхідний для атаки час. Багаторазове гешування може використовуватися і разом з додатковою випадковою величиною, що робить атаку ще більш складно здійсненною.

Також можна використовувати аналогічний метод, що полягає в гешування пароля специфічним алгоритмом (в роботі ми використовували функцію SCRYPT), в якому багато разів виконується складне перетворення. В якості параметрів алгоритм використовує не тільки пароль і випадкову величину, а й додатковий параметр – «вартість», який визначає число прогонів функції. Даний додатковий параметр дозволяє налаштовувати систему залежно від вимог безпеки, а також легко збільшувати необхідні для атаки ресурси в міру прогресу в розвитку обчислювальної техніки. Основною з вимог, що висуваються до таких функцій є складність (вимоги до пам’яті та часу) обчислення для того щоб ускладнити атаку грубою силою та отримання частин персональних даних користувачів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. S. F. Mjølsnes, S. Mauw, and S. K. Katsikas, Eds., “Public Key Infrastructure,” Lecture Notes in Computer Science, 2008.
2. D. Chadwick and G. Zhao, Eds., “Public Key Infrastructure,” Lecture Notes in Computer Science, 2005.
3. J. Lopez, P. Samarati, and J. L. Ferrer, Eds., “Public Key Infrastructure,” Lecture Notes in Computer Science, 2007.
4. A. Maeda, “PKI Solutions for Trusted E-Commerce: Survey of the De Facto Standard Competition in PKI Industries,” Information Technology Policy and the Digital Divide.
5. A. S. Atzeni and A. Lioy, Eds., “Public Key Infrastructure,” Lecture Notes in Computer Science, 2006.
6. J. Davies, “Implementing SSL/TLS Using Cryptography and PKI,” Dec. 2010.
7. K. Ohta and K. Koyama. Meet-in-the-Middle Attack on Digital Signature Schemes. In Abstract of AUSCRYPT ’90, pages 110-121, 1990.
8. RFC 5280: Internet X.509 Public Key Infrastructure - Certificate and Certificate Revocation List (CRL) Profile [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc5280.
9. RFC 4158: Internet X.509 Public Key Infrastructure - Certification Path Building. [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc4158.
10. RFC 6960: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc6960.
11. S. Farrell, "Not Reinventing PKI until We Have Something Better," in IEEE Internet Computing, vol. 15, no. 5, pp. 95-98, Sept.-Oct. 2011.

12. Krasnobayev V. A. Method for realization of transformations in public-key cryptography. Telecommunications and Radio Engineering. - Volume 66, 2007 Issue 17, pp. 1559-1572.
13. W. T. Polk and K. Seamons, “6th annual PKI R&D workshop ‘Applications-Driven PKI’ proceedings,” September 2007.
14. K. Isirova and O. Potii, "Decentralized public key infrastructure development principles," 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), Kiev, 2018, pp. 305-310.
15. B. Rajendran, "Evolution of PKI ecosystem," 2017 International Conference on Public Key Infrastructure and its Applications (PKIA), Bangalore, 2017, pp. 9-10.
16. H. Nanang, A. F. Misman and Z. Zulkifli, "Trust, risk and public key infrastructure model on e-procurement adoption," 2017 5th International Conference on Cyber and IT Service Management (CITSM), Denpasar, 2017, pp. 1-6.
17. Search International and National Patent Collections [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patentscope.wipo.int/search/en/result.jsf?_vid=P10-K48N6V-36076
18. Identity and access management [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.ibm.com/uk-en/security/identity-access-management
19. Microsoft identity platform [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developer.microsoft.com/en-us/identity
20. AWS Identity and Access Management (IAM) [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://aws.amazon.com/ru/iam/
21. Oleksandr Kurbatov, Pavel Kravchenko, Nikolay Poluyanenko, Oleksiy Shapoval, Tetiana Kuznetsova. "Decentralized Identification and Certification System." International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), Kharkiv, 2019.

22. P. Kravchenko, B. Skriabin, O. Kurbatov. Engineer’s Guide to Financial Internet – Kharkiv: Promart, 2019. – 247 c.
23. P. Landrock, "PKI, past, present and future," 2005 The IEE Seminar on Quantum Cryptography: Secure Communications for Business (Ref. No. 2005/11310), London, 2005, pp. 0_12-2/17.
24. M. Pala, S. Cholia, S. A. Rea and S. W. Smith, "Extending PKI Interoperability in Computational Grids," 2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID), Lyon, 2008, pp. 645-650.
25. X.1254: Entity authentication assurance framework – ITU [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.itu.int/rec/T-REC-X.1254.
26. Using OAuth 2.0 to Access Google APIs [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developers.google.com/identity/protocols/OAuth2
27. Manually Build a Login Flow [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developers.facebook.com/docs/facebook-login/manually-build-a-login-flow/
28. Oauth with the Twitter APIs [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developer.twitter.com/en/docs/basics/authentication/overview/oauth
29. Authorizing OAuth Apps [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://developer.github.com/apps/building-oauth-apps/authorizing-oauth-apps/
30. The OAuth 2.0 Authorization Framework [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc6749
31. Bitcoin: A Peer-to-Peer Electronic Cash System [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://bitcoin.org/bitcoin.pdf

32. Кравченко П. Блокчейн і децентралізовані системи: навч. посібник для студ. закладів вищ. освіти: у 3 частинах. Ч.1 П.Кравченко, Б. Скрябін, О. Дубініна, – Харків: ПРОМАРТ, 2018. – 400 с.
33. Coded Merkle Tree: Solving Data Availability Attacks in Blockchain [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://eprint.iacr.org/2019/1139.pdf
34. General Data Protection Regulation [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://gdpr-info.eu
35. Online Browsing Platform (OBP) [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.iso.org/obp/ui/#iso:std:iso-iec:24760
36. Systems and methods for an incremental, reversible and decentralized biometric identity management system [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/US10078758B1/
37. Systems and methods for distributed identity verification [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/US10237259B2/
38. Open ID Connect Authentication With OAuth2.0 Authorization [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://dzone.com/articles/open-id-connect-authentication-with-oauth20-author
39. What problems does OpenID Connect solve? [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.onelogin.com/pages/openid-connect
40. Session Hijacking [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/session-hijacking/
41. Identity management [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/US20110010762A1/
42. A. Menezes, P. van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. Chapter 9. CRC Press, 1996.

43. Method, apparatus, and computer program product for providing a group based decentralized authorization mechanism [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/WO2009133419A1/

44. Distributed identities [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/US7512649B2/

45. B. Schneier, “Applied Cryptography, Second Edition,” John Wiley & Sons, Inc. Oct. 2015.

46. I. M. Rodiana, B. Rahardjo and W. Aciek Ida, "Design of a Public Key Infrastructure-based Single Ballot E-Voting System," 2018 International Conference on Information Technology Systems and Innovation (ICITSI), Bandung - Padang, Indonesia, 2018, pp. 6-9.

47. Protection of confidentiality, privacy and financial fairness in a blockchain based decentralized identity management system [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://patents.google.com/patent/US20190182035A1/en?oq=US+2019182035+(A1)

48. V. Dolgov and I. Ishchenko, "Proposals of using chameleon-signature in Ukrainian prototype of combined PKI," 2010 International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET), Lviv-Slavske, 2010, pp. 303-303.

49. N. Ferguson, B. Schneier, and T. Kohno, “Cryptography Engineering,” Oct. 2015.

50. A. A. Kuznetsov, Yu. I. Gorbenko, D. I. Prokopovych-Tkachenko, M. S. Lutsenko, M. V. Pastukhov. “NIST PQC: Code-Based Cryptosystems.” Telecommunications and Radio Engineering, Volume 78, 2019, Issue 5, pp. 429-441. DOI: 10.1615/TelecomRadEng.v78.i5.50.

51. A. Kuznetsov, A. Pushkar'ov, N. Kiyan and T. Kuznetsova, "Code- based electronic digital signature," 2018 IEEE 9th International Conference on Dependable

Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 2018, pp. 331-336. DOI: 10.1109/DESSERT.2018.8409154.

52. I. Gorbenko, M. Yesina and V. Ponomar, "Anonymous electronic signature method," 2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), Kharkiv, 2016, pp. 47-50.
53. O. Potii, Y. Gorbenko and K. Isirova, "Post quantum hash based digital signatures comparative analysis. Features of their implementation and using in public key infrastructure," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 105-109.
54. Practical Byzantine Fault Tolerance [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://pmg.csail.mit.edu/papers/osdi99.pdf
55. David Mazieres. The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.stellar.org/papers/stellar-consensus-protocol.pdf.
56. Горбенко Ю.І. Інфраструктури відкритих ключів. Електронний цифровий підпис. Теорія та практика : монографія / Ю.І. Горбенко, І.Д. Горбенко ; Харк. нац. ун-т радіоелектрон., ЗАТ Інт інформ. технологій. – Х. : Форт, 2010. – 593 с.
57. R. Anderson. The classification of hash functions. Proc. of the IMA Conference on Cryptography and Coding, Cirencester, December 1993, Oxford University Press, 1995, pp. 83-95.
58. ANSI X9.30 (PART 2), “American National Standard for Financial Services – Public key cryptography using irreversible algorithms for the financial services industry – Part 2: The secure hash algorithm (SHA)”, ASC X9 Secretariat – American Bankers Association, 1993.
59. B. Preneel. Analysis and Design of Cryptographic Hash Functions. PhD thesis, Katholieke University Leuven, January 1993.

60. J.-J. Quisquater and J.-P. Delescaille. How Easy is Collision Search. New results and applications to DES. In Advances in Cryptology, Proceedings of CRYPTO’89, pages 408-415, 1990.
61. E. Biham. On the Applicability of Differential Cryptoanalysis to Hash Functions. In E.I.S.S Workshop on Cryptographic Hash Functions, pages 25-27, March 1992.
62. E. Biham and A. Shamir. Differential cryptoanalysis of FEAL and N-Hash. In Advances in Cryptology – Eurocrypt ’91, pages 1-16, 1991.
63. [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://www.cryptonessie.org.
64. A. Kuznetsov, I. Svatovskij, N. Kiyan and A. Pushkar'ov, "Code-based public-key cryptosystems for the post-quantum period," 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkov, 2017, pp. 125-130. DOI: 10.1109/INFOCOMMST.2017.8246365.
65. SIMPLE MAIL TRANSFER PROTOCOL [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc821
66. How Many Email Users Are There? [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://99firms.com/blog/how-many-email-users-are-there/#gref
67. G. Guo, J. Zhang, and J. Vassileva, “Improving PGP Web of Trust through the Expansion of Trusted Neighborhood,” 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Aug. 2011.
68. D. Wueppelmann, “PGP Auth: Using Public Key Encryption for Authentication on the Web.” A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of master of computer science. Ottawa, Ontario September, 2015.

69. K. Portz, J. M. Strong, and L. Sundby, “To Trust Or Not To Trust: The Impact Of WebTrust On The Perceived Trustworthiness Of A Web Site,” Review of Business Information Systems (RBIS), vol. 5, no. 3, p. 35, Jul. 2011.

70. M. Zhu and Z. Jin, “Trust Analysis of Web Services Based on a Trust Ontology,” Lecture Notes in Computer Science, pp. 642–648.

71. J. Callas, OpenPGP Message Format [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: www.ietf.org/rfc/rfc4880.txt.

72. Yu.V.Stasev, A.A.Kuznetsov, “Asymmetric Code-Theoretical Schemes Constructed with the Use of Algebraic Geometric Codes.” Cybernetics and Systems Analysis, vol. 41, Issue 3, pp. 354-363, May 2005. DOI: 10.1007/s10559-005-0069-9.

73. A. Kuznetsov, M. Lutsenko, N. Kiian, T. Makushenko and T. Kuznetsova, "Code-based key encapsulation mechanisms for post- quantum standardization," 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 2018, pp. 276-281. DOI: 10.1109/DESSERT.2018.8409144.

74. Yu.V. Stasev, A.A. Kuznetsov. “Asymmetric code-theoretical schemes constructed with the use of algebraic geometric codes”. Kibernetika i Sistemnyi Analiz, No. 3, pp. 47-57, May-June 2005.

75. A. Kuznetsov, A. Kiian, M. Lutsenko, I. Chepurko and S. Kavun, "Code based cryptosystems from NIST PQC," 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 2018, pp. 282-287. DOI: 10.1109/DESSERT.2018.8409145.

76. Merkle, R. C. (1988). "A Digital Signature Based on a Conventional Encryption Function". Advances in Cryptology — CRYPTO '87. Lecture Notes in Computer Science. 293. pp. 369–378. doi:10.1007/3-540-48184-2_32.

77. Закону України Про захист персональних даних від 01.06.2010 № 2297-VI [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/2297-17

78. Словникові атаки на геш-функції [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://www.panasenko.ru/Articles/168/168.html

79. Курбатов О.С., Кравченко П.О., Полуяненко М.О., Шаповал О.В., Кузнецова Т.Ю. (2019) ДЕЦЕНТРАЛІЗОВАНА СИСТЕМА ІДЕНТИФІКАЦІЇ ТА СЕРТИФІКАЦІЇ Кібербезпека, освіта, наука, техніка, 2(5), 16-30