УДК 004.056

ВИКОРИСТАННЯ ТЕХНОЛОГІЇ BLOCKCHAIN ДЛЯ ІДЕНТИФІКАЦІЇ

А. П. Гуліта[1, a], А. М. Родіонов[1]

[1] *Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Фізико-технічний інститут*

Анотація

Архітектура відкритих ключів (PKI), що широко використовується ідентифікації ідентичностей у вигляді їх прив'язки до відкритих ключів, має певні недоліки, спричинені централізацією. У даній роботі пропонується альтернативний підхід до побудови PKI, а саме її децентралізація. Децентралізована архітектура відкритих ключів (DPKI) забезпечує незалежність від довіреної третьої сторони, такої як центр сертифікації (CA), що нівелює проблему єдиної точки відмови. Запропонована DPKI побудована на децентралізованому прозорому дизайні, використовуючи Blockchain технологію, а саме розумні контракти на платформі Ethereum, робить перевірку сертифікатів тривіальною. На відміну від централізованої моделі верифікації ідентичності, запропонована модель побудована таким чином, що користувач може підтвердити певні атрибути ідентичності іншого користувача, підписавши їх відповідним сертифікатом.

Ключові слова: ідентифікація, архітектура відкритих ключів, децентралізована система, технологія Blockchain, платформа Ethereum

Вступ

Доказ ідентичності часто широко використовується, зокрема, наприклад, коли хост мережі має на меті аутентифікувати себе для того, щоб встановити безпечне з'єднання з іншим хостом. Кожен хост володіє парою відкритий-закритий ключ, і для того, щоб встановити безпечне з'єднання вони обмінюються відкритими ключами. При цьому виникає необхідність підтвердити, що відкритий ключ дійсно належить пред'явнику. PKI використовується для цієї мети. При цьому публічний ключ асоціюється з відповідною ідентичністю через реєстрацію і видачу відповідних сертифікатів за участі централізованих CA. PKI складається з набору ролей та процедур створення, управління та відклику цифрових сертифікатів.

У даній роботі пропонується децентралізована архітектура відкритих ключів, яка базується на Web-of-Trust моделі, у якій не використовуються CA, а натомість є можливість для користувача підписати публічний ключ іншого, для засвічення його автентичності.

Модель Web-of-Trust є кроком на шляху до децентралізованої PKI, однак не є PKI, оскільки не передбачає механізму відклику сертифікатів.

Цифровий сертифікат містить публічний ключ разом з іншою інформацією, такою як дата закінчення дії сертифікату та ім'я власника. Таким чином існує зв'язок між іменем користувача і його відкритим ключем [1]. Поширеним стандартом для PKI є X.509 [2]. Додаток до стандарту X.509 містить список потенційно можливих атрибутів для сертифікату, таких як назва установи, поштовий індекс чи електронна адреса. Поєднання ключової мети стандарту X.509 та моделі Web-of-Trust відкриває можливість рішення проблем ідентифікації у контексті, коли особа чи організація має на меті підтвердити певний атрибут ідентичності іншої особи.

Для прикладу, атрибути університетської освіти може підписати довірена особа університету, яка за це відповідає, а наявність страхового полісу – відповідна страхова установа. X.509 сертифікат є конгломератом з відкритого ключа користувача, інформації користувача (атрибутів), імені сертифікату та цифрового підпису CA, який скріплює всі ці дані воєдино [3]. Тобто, CA підписує весь сертифікат загалом, з усіма його атрибутами. Натомість у роботі пропонується підхід з індивідуальним підписанням атрибутів користувачами системи.

Для побудови зазначеної системи використовується платформа Ethereum і однією з переваг такої реалізації DPKI є те, що розподілена blockchain-база сертифікатів є незаперечною та забезпечує прозорість і швидкість перевірки й відклику сертифікатів, і має властивість моментального поширення між усіма користувачами [4].

1. Технологічне підґрунтя та ідеологія

1.1. Blockchain

Принципова новизна технології Blockchain полягає в її архітектурі, що забезпечує можливість виконання децентралізованих транзакцій, без використання будь-яких посередників. Замість того, щоб встановлювати і підтримувати довірчі відносини з партнером по транзакції або стороннім учасником-

[a] aniahulita@gmail.com

посередником, користувачі покладаються на загальнодоступну розподілену базу даних, збережену на децентралізованих вузлах [5].

На кожному вузлі, тобто комп'ютері, підключеному до мережі за допомогою клієнта, що виконує перевірку і передачу транзакцій, зберігається копія блокчейна, яка автоматично завантажується, у момент підключення майнера до блокчейн-мережі. В реєстрі зберігається повна інформація про всі адреси і баланси, починаючи з генезис-блоку (найпершого блоку транзакцій) до останнього добавленого блоку.

Блок транзакцій – спеціальна структура для запису групи транзакцій, який містить хеш заголовку та інформацію про попередній блок. Усі блоки можна вибудувати в один ланцюг, який містить інформацію про всі виконані операції в системі.

1.2. Ethereum та розумні контракти

Ethereum є платформою, що дозволяє реалізовувати децетралізовані додатки з використанням технології Blockchain, наділені властивостями масштабованості та Тьюринг-повноти [6]. Віртуальна машина Ethereum – головна інновація проекту Ethereum. Ця віртуальна машина розроблена таким чином, що вона виконується усіма учасниками однорангової мережі, яка може читати і записувати до блокчейну як виконуваний код так і дані, перевіряти цифрові підписи і виконувати код у Тьоюринг-повній манері [7]. Код буде виконуватись на віртуальній машині лише за умови отримання підтвердженого через цифровий підпис повідомлення.

Ethereum може бути альтернативно описаний як блокчейн з вбудованою мовою програмування, або як глобально виконувана віртуальна машина що базується на досягненні консенсусу. Частиною протоколу, що фактично обробляє внутрішні стани і виконує обчислення є віртуальна машина Ehtereum (EVM).

У Ethereum стан мережі визначається об'єктами, які називаються акаунтами, кожен з яких має 20-байтову адресу, та станом транзакцій, які безпосередньо передають дані та інформацію між акаунтами.

Розумні контракти Ethereum визначаються як комплексні додатки, що включають в себе володіння цифровими активами, які безпосередньо контролюються правилами, що імплементовані за допомогою коду.

Розумні контракти на Ethereum написані низькорівневою stack-based bytecode мовою, які виконуються віртуальною машиною Ethereum і є EVM-кодом. Розумні контракти також можуть бути написані на високорівневій мові, такій як Serpent або Solidity, а потім бути скомпільовані у EVM-код.

Документація Ethereum визначає декілька потенційно можливих способів використання розумних контрактів, один з яких – це системи ідентифікації користувачів [8]. Наприклад, розумний контракт може бути створений для зв'язку доменного імені з IP-адресою для побудови децентралізованої реєстраційної системи. NameCoin є прикладом такої системи, що базується на BitCoin Blockchain.

Ключовим завданням даної роботи є написання таких розумних контрактів які б відповідали функціональності операцій архітектури відкритих ключів і системи ідентифікації. У даній роботі пропонується систему, де публічний ключ і атрибути користувача зберігаються поверх блокчейнів і контролюються за допомогою розумних контрактів.

1.3. Web-of-Trust

Ключовою імплементацією PKI, як описано в стандарті X.509 є використання сертифікатів і CA. Вона найкраще підходить для структурованих організаційних ієрархій з довіреним органом, який підтверджує всі видані сертифікати [9].

Альтернативою моделі довірених центрів сертифікації є Web-of-Trust. Концепція Web-of-Trust була вперше описана 1992 року в документації PGP 2.0 і звучить наступним чином: "з плином часу, ви будете накопичувати ключі від інших користувачів, яких ви можете визначити як довірених поручителів. Кожен може визначити своїх індивідуальних довірених поручителів. Користувач буде накопичувати і поширювати разом з публічним ключем набір сертифікатів від інших людей, очікуючи, що інший користувач довірятиме, принаймні одному або двом з сертифікатів. Це призведе до появи децентралізованої відмовостійкої мережі довіри для всіх відкритих ключів" [10].

У моделі Web-of-Trust немає центрів сертифікації. Натомість будь-який користувач системи може підписати публічний ключ іншого, що означає відсутність центру сертифікації в системі, як великої точки відмови, оскільки за дизайном моделі публічний ключ може мати декілька підписів. Це означає, що якщо певний підписувач сертифікату є скомпрометованим і його ключ є відкликаним (revoked), – то вплив цього інциденту на мережу є обмеженим.

2. Модель розробленої DPKI

Характеристики, якими володіє PKI, реалізовано на Blockchain інфраструктурі, а саме:

- механізм сертифікації атрибутів, якими характеризується користувач, асоційованими з його публічним ключем,
- механізм відклику наданих сертифікатів,
- можливість контролю доступу до атрибутів.

2.1. Модель взаємодії

Користувач який володіє парою відкритий-закритий ключ, створює і контролює розумний контракт, який містить атрибути (наприклад ім'я, вік, ідентифікаційний код тощо). Список атрибутів не є фіксованим, і може варіюватись за бажанням користувача. Інші учасники взаємодії мають можливість вказати на достовірність представлених атрибутів, підписавши відповідні сертифікати, або ж запросити сертифікати, які уже підписані. На рис. 1 зображено модель взаємодії в системі ідентифікації.

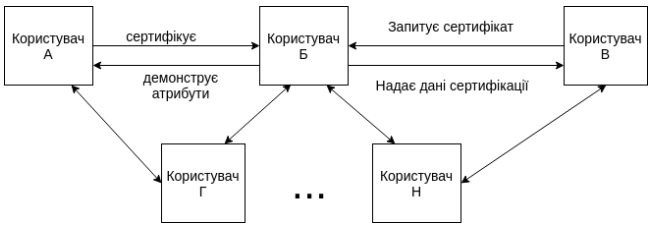

Рис. 1. Модель взаємодії користувачів

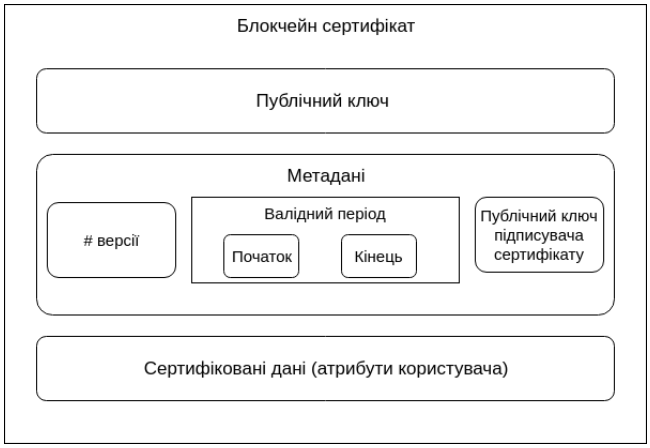

Рис. 2. Блокчейн сертифікат

2.2. Blockchain сертифікат

Blockchain PKI використовує Blockchain сертифікат. Сертифікат (рис. 2) скріплює воєдино публічний ключ, метадані, та дані які підлягають сертифікації.

Щоб визнати дійсним блокчейн сертифікат, верифікатор перевіряє, що сертифікат присутній у сховищі підписаних сертифікатів користувача і відсутній у серед відкликаних сертифікатів цього користувача.

3. Імплементація

3.1. Сутності даних

Представлена система базується на Ethereum блокчейні та контролюється розумними контрактами, які у відповідності до моделі Web-of-Trust надають користувачам можливість керування (збереження, відтворення та перевірки) власної ідентичності та атрибутів ідентичності інших осіб.

На рис. 3 представлено сутності даних і їх взаємозв'язки, які визначаються у контрактах.

Розумний контракт сконцентрований навколо користувача, який публікує набір атрибутів, підписів та відкликаних сертифікатів на блокчейні для демонстрації своєї ідентичності. Користувач асоціюється з Ethereum-адресою, що виступає в якості відкритого ключа та контролюється приватним ключем.

Мова написання контрактів Solidity підтримує реалізацію struct, для задання структур даних, що використовується у реалізації. Entity (користувач) фактично не визначається як структура даних у контракті, проте вона обумовлена самою реалізацією Ethereum. Кожна Ethereum транзакція має свого власника, який є 20-байтною адресою. Для того, щоб транзакція провалідувалась, потрібно щоб вона була

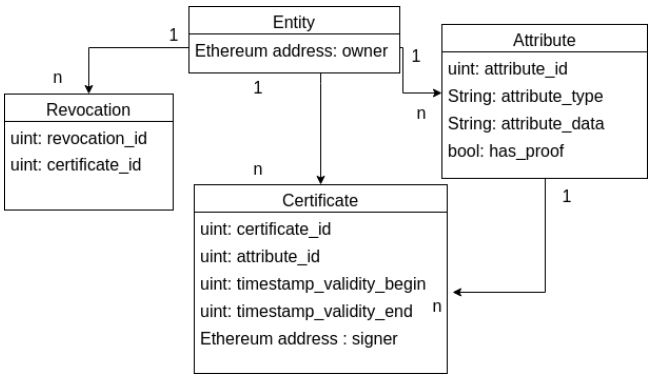

Рис. 3. Діагарма взаємлзв'язків сутностей для Entity контракту

підписана приватним ключем, асоційованим з адресою власника. Це робить Ethereum адресу хорошим представленням особи чи користувача системи.

Структури Attribute, Sertificate та Revocation мають поле адресу користувача. Для Sertificate поле користувача визначає підписувача сертифікату. Для Revocation (відклику сертифікатів) – користувач існує за замовчуванням, тому що здійснити відклик сертифікату може тільки користувач, який його надав.

3.2. Алгоритм відклику сертифікатів

Функція revokeSignature() дозволяє користувачу здійснити відклик сертифікату, який ним же був наданий. Параметр який приймає функція – це посилання на сертифікат, який потрібно відкликати. Функція зберігає всі відкликані контракти у масив, який є доступним іншим користувачам.

Відклик сертифікату відбувається лише в тому випадку, якщо адрес користувача який викликає функцію revokeSignature() співпадає з адресою користувача, який надав сертифікат. Процедуру відклику сертифікату наведено в Алгоритмі 1.

Algorithm 1: Процедура відклику сертифікату

```
Input parameters: :certificate_id
Result: certificate is revoked
Function revokeSignature
    if transaction.sender =
     certificates[certificate_id].signer then
        revocation_id ← revocations.length++
        revocation ← revocations[revocation_id]
        revocation.certificate_id ← certificate_id
        return successful
    else
        return failed
    end
```

3.3. Алгоритм надання сертифікатів

За допомогою функції signAttribute() виконується надання сертифікату атрибуту, переданому в якості аргументу функції. Функція зберігає наданий

сертифікат у масив сертифікатів користувача. Виконання функції завершується успішно за умови, якщо власник атрибутів надав відповідний дозвіл підписувачу, тобто якщо адрес підписувача сертифікату є в списку адрес, що мають доступ на перегляд та підпис. Процедуру надання сертифікату наведено в Алгоритмі 2.

Algorithm 2: Процедура надання сертифікату

```
Input parameters:: attribute_id,
                   timestamp_validity_end
Result: certificate is added
Function revokeSignature
    if hasAccess[transaction.sender] = true then
        certificate_id ← certificates.length++
        certificate ← certificates[certificate_id]
        certificate.signer ← transaction.sender
        certificate.attribute_id ← attribute_id
        certificate.timestamp_validity_end ←
        timestamp_validity_end
        return successful
    else
        return failed
    end
```

3.4. Політика доступу до атрибутів

Розроблений контракт Owned визначає політику доступу до особистих даних, що описується наступними правилами:

- В контракті задається власник, тобто користувач, що створив контракт, який має право модифікації та перегляду особистих даних.
- Власник контракту має можливість надавати іншим вибраним користувачам право перегляду його особистих даних та надання відповідних сертифікатів.
- Для попередження проблеми, коли власник контракту втратив свій приватний ключ, і не має доступу для модифікації свої даних, в контракті "Owned"передбачена можливість делегувати права власника іншому користувачеві.

Висновки

У даній роботі наведено розробку децентралізованої архітектури відкритих ключів, яка використовує прозорість технології Blockchain та володіє принципами управління атрибутами у відповідності до моделі Web-of-Trust. Реалізацію було виконано у формі розумних контрактів платформи Ethereum.

Адаптивність користувачів до розробленої системи включає той фактор, що для її використання дві сторони взаємодії повинні бути її користувачами. Наприклад, якщо університет бажає підписати науковий ступінь користувача системи, користувач спершу повинен додати такий атрибут до списку своїх атрибутів. Певною мірою це збільшує бар'єр на шляху адаптації до представленої системи. Проте таким чином користувач має повний контроль над тим, які атрибути представляють його ідентичність, а які ні.

У роботі наведено модель взаємодії користувачів у системі, наведено алгоритми для забезпечення функцій архітектури відкритих ключів та окреслено переваги такої реалізації архітектури.

Перелік використаних джерел

1. Венбо М. Современная криптография. Теория и практика. — Издательство Вильямс, 2005.
2. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile / D. Cooper, S. Santesson, S. Farrell et al. — 2008. — Access mode: https://tools.ietf.org/html/rfc5280.
3. RDIG Certification Authority. — Access mode: http://ca.grid.kiae.ru/RDIG/info/x509.html.
4. Равал С. Децентрализованные приложения. Технлогия Blockchain в действии. — O'Reilly Media Inc, 2017.
5. Swan Melanie. Blockchain. Blueprint for a New Economy. — O'Reilly Media Inc, 2015.
6. Ethereum Development Tutorial. — Access mode: https://github.com/ethereum/wiki/wiki/Ethereum-Development-Tutorial.
7. Swanson T. Great Chain of Numbers: A Guide to Smart Contracts, Smart Property and Trustless Asset Management. — Texas A&M University, 2014.
8. A Next-Generation Smart Contract and Decentralized Application Platform. — Access mode: https://github.com/ethereum/wiki/wiki/White-Paper.
9. Н.Фергюсон, Шнайер Б. Практическая криптография. — Компьютерное издательство Диалектика, 2005.
10. Nemati H. R. Applied Cryptography for Cyber Security and Defense: Information Encryption and Cyphering. — Information Science Reference. New York, 2010.