УДК 004.738.5 DOI: 10.26565/2519-2310-2019-1-04

ЗАСТОСУВАННЯ КРИПТОАЛГОРИТМІВ В ДЕЦЕНТРАЛІЗОВАНИХ МЕРЕЖАХ ТА ПЕРСПЕКТИВИ ЇХ ЗАМІНИ ДЛЯ ПОСТКВАНТОВОГО ПЕРІОДУ

Олександр Якішин, Олександр Оникійчук, Володимир Скриннік, Катерина Кузнецова

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
dkrakaslian@gmail.com, onik4524a@gmail.com, v.skrynnik@karazin.ua, kate7smith12@gmail.com

Рецензент: Олександр Оксіюк, д.т.н., проф., Київський національний університет імені Т. Шевченка, вул. М. Ломоносова 81, Київ, 03189, Україна.
o.oksiuk@gmail.com

Надійшло: Березень 2019.

Анотація: *В роботі проведено огляд використовуваних у блокчейн системах електронних підписів та функцій хешування. Наведено криптографічні алгоритми, які використовуються або можуть використовуватися в децентралізованих мережах. Представлені алгоритми для захисту, як від класичних, так і від квантових атак. Результати роботи свідчать про необхідність вдосконалення криптографічних алгоритмів для захисту від можливих квантових атак в майбутньому.*

Ключові слова: *криптографічні алгоритми; алгоритми захисту децентралізованих систем; блокчейн.*

1 Вступ

Нині досить широке розповсюдження знаходять технології блокчейн, в цих технологіях особливо жорсткі вимоги висувають до надання користувачам таких послуг безпеки, як цілісність, справжність, доступність. Ці вимоги задовольняються використовуючи у цих системах алгоритм електронного підпису (ЕП) та функції гешування (ФГ). ЕП у блокчейні, в основному, використовується для підпису транзакцій та, у окремих випадках (таких як Etherium), для виконання смарт-контракту, при цьому основними вимогами до електронного підпису є забезпечення кріптостійкості проти класичних та квантових атак. Одночасно, сутність використання ФГ полягає у застосуванні її до обчислення геш значень блоків та зв'язування ланцюгів блоків, при цьому основними вимогами до геш-функцій є забезпечення криптостійкості проти класичних та квантових атак, а також мінімізації складності (підвищення швидкості) гешування. Метою цієї роботи є загальний аналіз вже використовуваних у блокчейн системах ЕП та ФГ і огляд перспективних ЕП та ФГ, які у майбутньому можуть їх замінити.

2 Визначення блокчейн

Блокчейн (*англ. Blockchain, Block chain від block – блок, chain – ланцюг*) [1] – це незмінні системи цифрових регістрів, які реалізовані розподіленим чином (тобто, без центрального місця сховища) та зазвичай без центрального органу управління. На самому базовому рівні вони дозволяють користувачам записувати транзакції в загальнодоступному реєстрі, так що ніяка транзакція не може бути змінена після її опублікування.

Транзакція - це запис передачі активів (цифрова валюта, одиниці запасів, тощо) між сторонами. Аналогом цього було б запис на перевірочному рахунку кожного разу коли гроші було депоновано або вилучено. У Таблиці 1 наведено умовний приклад транзакції. Кожен блок в блокчейні містить кілька транзакцій. Одна транзакція зазвичай вимагає принаймні наявність наступних інформаційних полів, але може містити більше:

• Сума - загальна сума переданого цифрового активу.

• Вхід - список цифрових активів, що підлягають передачі (їх загальна вартість дорівнює сумі). Слід зверніть увагу на те, що кожен цифровий актив ідентифікується унікально і може мати різні значення інших активів. Однак при цьому активи не можуть бути додані або вилучені з існуючих цифрових активів. Замість цього, цифрові активи можуть бути розділені на

кілька нових цифрових активів (кожен з меншою вартістю) або об'єднані, щоб сформувати меншу кількість нових цифрових активів (кожен з відповідним чином більшим значенням).

• Виходи - рахунки, які будуть одержувачами цифрових активів. Кожен вихід вказує значення, яке буде передано новому власнику (власникам), особу нового власника (ів), і набір умов, яким повинні відповідати нові власники, щоб отримати цю вартість. Якщо наданих цифрових активів більше, ніж потрібно, додаткові кошти повертаються відправнику (це механізм “внесення змін”).

ID транзакції / Hash - унікальний ідентифікатор кожної транзакції. Деякі блокчейн використовують ідентифікатор, а інші приймають геш конкретної транзакції, як унікальний ідентифікатор.

Таблиця 1 - Приклад транзакції

ID транзакції / Hash	Вхід	Виходи	Вага	Сума
Ідентифікатор транзакції: 0xa1b2c3	Рахунок A	Рахунок B	0.0321	
		Рахунок C	2.5000	2.5321

Важливим є визначення дійсності угоди (*той факт, що хтось претендує на те, щоб угода відбулася, ще не означає, що вона дійсно сталася*). Транзакції підписуються і їх можна перевірити за допомогою пар публічних/приватних ключів у будь-який час.

3 Геш

Важливою складовою технології блокчейн є використання для багатьох операцій криптографічних геш-функцій, наприклад таких як гешування змісту блоку. Гешування - це метод обчислення відносно унікального виводу фіксованого розміру (називається дайджест повідомлення або просто дайджест) для входу майже будь-якого розміру (наприклад, файлу, тексту або зображення). При цьому, навіть найменша зміна вхідного значення (наприклад, одного біта) призведе до абсолютно іншого дайджесту виводу (див. Табл. 2).

В цілому геш-алгоритми розроблені так, щоб вони були односторонніми (відомі як стійкі до показу): це означає, що неможливо знайти обчислювальну інформацію для будь-якого входу, який відображається на будь-якому заздалегідь заданому виході. Якщо треба знайти бажаний конкретний висновок, багато входів повинні бути випробувані шляхом передачі їх через геш-функцію, поки не буде знайдено вхідний сигнал, який дасть бажаний результат. Крім того, геш алгоритми розроблені таким чином, щоб бути стійкими до зіткнень (відомі як другий стійкий до показу).

Таблиця 2 - Приклади вхідного значення та SHA-256 вихідного значення

Вхідне значення	SHA-256 вихідне значення
1	0x6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b
2	0xd4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35
Hello, World!	0xdffd6021bb2bd5b0af676290809ec3a53191dd81c7f70a4b28688a362182986f

Наприклад алгоритм гешування, що використовується у багатьох технологіях блокчейн, є Secure Hash Algorythm (SHA) з вихідним розміром 256 біт (SHA-256). Багато комп'ютерів апаратно підтримують цей алгоритм, що робить дуже зручним для обчислення.

Оскільки ФГ має дуже велику кількість можливих вхідних значень та безкінечну кількість вихідних значень, то можливо знайти колізію де $hash(x) = hash(y)$. Однак на даний момент це малоймовірно тому, що будь-який такий вхід x та y, який виробляє один і той самий дайджест, був би дійсним у контексті системи блокчейн (в даному випадку це були би обидві дійсні транзакції блокчейн), а також обчислювався б досить близько один до одного по часу. У переважної більшості сучасних комп'ютерів занадто мала обчислювальна швидкість для роботи з існуючими алгоритмами, що націлені на виконання подібних задач. В цьому контексті стоїть відмітити, що ще з 1980-го року почалась розробка квантового комп'ютера, який повинен значно перевершити по своїй обчислювальної потужності традиційні комп'ютери за допомогою обчислень що базуються на законах квантової механіки. Згідно з даними інтерв'ю з менеджера компанії IBM® Марком Кетченом, створення квантового комп'ютера прогнозується через 3-8 років [2]. На сьогодні існують два алгоритма для квантового комп'ютера, це алгоритм Гровера [3], який вирішує проблему перебору, та алгоритм Шора [4] що вирішує проблему факторизації (*розкладу одного числа на два*). З цих двох квантових алгоритмів для пошуку колізій може використовуватись алгоритм Гровера. Алгоритм Шора, на даний момент, ніяк не загрожує ФГ.

Розглянемо найпоширеніші ФГ у блокчейн системах:

1. SHA-256 (англ. *Secure Hash Algorithm Version 2* – безпечний алгоритм гешування, вер. 2), це збірна назва односторонніх геш-функцій SHA-224, SHA-256, SHA-384 і SHA-512 [5]. Найпоширеніша ФГ завдяки своїй простоті обчислення (високій швидкодії) та доволі непоганою стійкістю до колізій та пошуку першого прообразу, та має один основний недолік це невелика у порівнянні з іншими найпоширенішими ФГ стійкість до знаходження другого прообразу, що зазвичай і стає причиною використання сторонніх ФГ на вихідне значення SHA-256. Наприклад, у Bitcoin вихідне значення SHA-256 гешується спочатку RIPMD-160, потім BASE-58 і вже після цього використовується у системі блокчейн. На даний момент SHA-256 використовується у таких блокчейн системах як: Bitcoin, Litecoin, Tether, Cornado (*HMAC-SHA512*) і т.д.

2. Keccak (SHA-3) – геш функція побудована за принципом криптографічної губки [6] (*дана структура криптографічних алгоритмів була запропонована авторами алгоритму Keccak раніше*). Він є переможцем останнього на сьогодні конкурсу NIST SHA-3, де відрізнився від конкурентних ФГ підвищеною стійкістю до усіх видів атак, але не найбільшою швидкодією. Оригінальний алгоритм Keccak має безліч параметрів, що налаштовуються з метою забезпечення оптимального співвідношення криптостійкості і швидкодії для певного застосування алгоритму на певній платформі. Регульованими величинами є: розмір блоку даних, розмір стану алгоритму, кількість раундів у функції *f()* та інші. На даний момент Keccak використовується у таких блокчейн системах як Ethereum, Nexus(NXS), Quark(QRK), SmartCash(SMART), Maxcoin(MAX), CreativeChain(CREA).

Таблиця 3 – Порівняння параметрів алгоритмів SHA-256 та Keccak (SHA-3)

Алгоритм	Розмір виводу (біт)	Розмір внутрішнього стану	Розмір блоку	Довжина розміру	Розмір слова	Раунди
SHA -224 -256	256	256	512	64	32	64
SHA-384, -512	384/512	512	1024	128	64	80
Keccak (SHA-3)	224/256/384 /512	1600	1600-2*bits (1152/1088/ 832/576)	–	64	24

Аналіз даних що наведені в Табл. 3 [7], дозволяє стверджувати, що кожний алгоритм має свій розмір виводу, що варіюється від 256 до 512 біт, а найбільший розмір внутрішнього ста-

ну має алгоритм Keccak (SHA-3) – 1600, довжина розміру алгоритму варіюється від 64 до 128, а розмір слова (в залежності від алгоритму) від 32 до 64.

З аналізу даних Табл. 4 [8] можна зробити висновок, що 512 версії алгоритмів є кращими за своїх попередників, оскільки мають найбільшу стійкість до: - колізій; - знаходження прообразу; - знаходження другого прообразу та до алгоритму Гровера. Найкращим можна назвати ФГ SHA-3 512, оскільки має найкращу стабільність та стійкість, а найгіршою - ФГ SHA-256.

Таблиця 4 – Порівняння стійкості ФГ (у бітах)

Функції гешування	Стійкість до колізій	Стійкість до знаходження прообразу	Стійкість до знаходження 2-го прообразу	Стійкість до кв. алгоритму Гровера
SHA-256	128	256	201-256	64
SHA-512	256	512	394-512	128
SHA-3 256	128	256	256	64
SHA-3 512	256	512	512	128

Важливим критерієм для порівняння ФГ є складність/швидкодія хешування [8], що наведені у Табл. 5.

Таблиця 5 - Результати тесту швидкодії ФГ

Алгоритм	MiB/sec	Циклів/байт
SHA-3 256	46	6,2
SHA-256	111	15,8
SHA-512	99	17,7
SHA-3 512	67	5,8

З даних Табл. 5 можна побачити, що алгоритм SHA-256 має найбільший показник MiB/sec – 111, SHA-3 256 – найменший, 46. ФГ SHA-512 виконує 17,7 циклів за байт, а SHA-3 512 найменше – 5,8. Таким чином, серед наведених алгоритмів, SHA-256 та SHA-512 – найшвидші ФГ.

В цілому за результатами аналізу даних всіх вище наведених таблиць можна зробити висновок, що сучасні функції гешування дозволяють забезпечити необхідний рівень стійкості проти усіх відомих класичних атак. Однак, варто зазначити, що з вхідною довжиною у 256 біт стійкість до алгоритму Гровера не перевищує 64 біт, що не має необхідної стійкості для постквантового періоду. Хоча за результатами тесту швидкості найкращим є SHA-2 варто не забувати, що наведені результати стосуються тільки програмної реалізації цих ФГ, швидкодія яких відрізняється від реалізації цих ФГ на мікроконтролерах, також SHA-2 у порівнянні з більшістю інших приведених функцій гешування відрізняється меншою стійкістю до атак знаходження другого прообразу.

4 Криптографія асиметричного ключа

Фундаментальною технологією, що використовується технологіями блокчейн, є *криптографія асиметричних ключів* (інша назва криптографія публічного/приватного ключа). Криптографія асиметричного ключа використовує пару ключів: відкритий ключ і приватний ключ, які математично пов'язані між собою. Відкритий ключ може бути оприлюднений без зниження безпеки процесу, але приватний ключ повинен залишатися таємним, якщо дані повинні зберігати криптографічний захист. Навіть при наявності взаємозв'язку між двома ключами, це не дає можливості ефективно визначити приватний ключ на основі знання відкритого ключа. Криптографія асиметричного ключа використовує різні ключі з пар ключів для конкретних функцій, залежить від того, яка послуга повинна бути надана. Наприклад, при цифро-

вому (електронному – ЕП) підпису даних, криптографічний алгоритм використовує приватний ключ для підпису. Потім підпис можна перевірити за допомогою відповідного відкритого ключа.

Використання криптографії асиметричного ключа в системах блокчейн:

• приватні ключі використовуються для цифрового підпису транзакцій;

• відкриті ключі використовуються для виведення адрес, що дозволяє використовувати підхід один-до-багатьох, що запроваджує псевдоанімність (одна пара відкритих ключів може дати кілька адрес);

• відкриті ключі використовуються для перевірки підписів, створених за допомогою приватних ключів;

• криптографія асиметричного ключа дає можливість перевірити, що користувач передає значення іншому користувачеві, що володіє приватним ключем, здатним підписати значення.

Деякі блокчейн системи, наприклад такі як Monero, використовують ЕП у протоколі кільцевого підпису. Кільцевий підпис - один з механізмів реалізації електронного підпису, при якому відомо, що повідомлення підписав один з членів списку потенційних підписантів, але не розкриває, хто саме. Підписант самостійно формує список з довільного числа осіб (включаючи і себе). Для накладання підпису підписувачу не потрібен дозвіл, сприяння або допомога з боку включених у список осіб, використовуються лише відкриті ключі всіх членів списку і власний закритий ключ.

Варто позначити, що у більшості алгоритмів ЕП криптостійкість базується на складності вирішення проблеми факторизації, що робить загрозою не тільки алгоритм Гровера, але й алгоритм Шора.

Розглянемо деякі найпоширеніші алгоритми ЕП застосовані у системах блокчейн [9].

ECDSA [10] - алгоритм з відкритим ключем для створення цифрового підпису, аналогічний за своєю будовою DSA, але визначений, на відміну від нього, не над кільцем цілих чисел, а в групі точок еліптичної кривої. Стійкість алгоритму шифрування ґрунтується на задачі дискретного логарифма в групі точок еліптичної кривої. На відміну від задачі простого дискретного логарифма і задачі факторизації цілого числа, не існує субекспоненціального алгоритму для задачі дискретного логарифма в групі точок еліптичної кривої. З цієї причини «сила на один біт ключа» істотно вище в алгоритмі, який використовує еліптичні криві. Крім того, ECDSA використовує не тільки алгебраїчні властивості еліптичних кривих, але і кінцеві поля. Кінцеве поле - це заданий діапазон додатних чисел, в рамках якого лежать результати алгебраїчних обчислень. Еліптичні криві в рамках кінцевого поля змінюються до невпізнання. Але, незважаючи на втрачену «красу», все математичні властивості кривої залишаються колишніми. Для підписування повідомлень необхідна пара ключів - відкритий і закритий. При цьому закритий ключ повинен бути відомий тільки тому, хто підписує повідомлення, а відкритий - будь-кому хто бажає перевірити справжність повідомлення. Також загальнодоступними є параметри самого алгоритму. ECDSA є дуже привабливим алгоритмом для реалізації цифрового підпису.

Найважливішою перевагою ECDSA є можливість його роботи на значно менших полях $F(p)$. Як, загалом, з криптографією еліптичної кривої, передбачається, що бітовий розмір відкритого ключа, який буде необхідний для ECDSA, дорівнює подвійному розміру секретного ключа в бітах. Для порівняння, при рівні безпеки в 80 біт (тобто атакуючому необхідно приблизно 2^{80} версій підписів для знаходження секретного ключа), розмір відкритого ключа DSA дорівнює, принаймні, 1024 біт, тоді як відкритого ключа ECDSA - 160 біт. З іншого боку розмір підпису однаковий і для DSA, і для ECDSA: $4 \cdot t$ біт, де t - рівень безпеки, який вимірюється в бітах, тобто - приблизно 320 біт для рівня безпеки в 80 біт. У мережі Bitcoin ECDSA використовується зі спеціальними параметрами означеними, як secp256k, secp256k1, він майже ніколи не використовувався до того, як Bitcoin став популярним, але зараз він набуває популярності завдяки своїм кількома приємним властивостям. Найбільш часто використовувані криві мають випадкову структуру, але secp256k1 був побудований спеціальним невипадковим способом, який дозволяє особливо ефективно обчислювати. Як наслідок, він ча-

сто більш ніж на 30% швидше, ніж інші криві, якщо реалізація достатньо оптимізована. Крім того, на відміну від популярних кривих NIST, константи secp256k1 були обрані передбачуваним способом, що значно зменшує можливість створення творцем кривої будь-якого типу бекдора. На сьогодні ECDSA використовується у більшості децентралізованих систем, таких як: Bitcoin, Ethereum, Litecoin та інші.

З даних таблиці 6 видно, наскільки алгоритм ЕП ECDSA кращий за DSA [11], тому що маючи однаковий рівень безпеки, розмір ключа алгоритму ECDSA набагато менший за свого конкурента, а це може стати перевагою у програмах, в яких є обмеження в реальному часі та пам'яті.

Таблиця 6 - Порівняння розміру ключа

Безпека (у бітах)	DSA – розмір ключа	ECDSA – розмір ключа
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

Так як у більшості алгоритмів ЕП криптостійкість базується на складності вирішення проблеми факторизації, що робить загрозою не тільки алгоритм Гровера, але й алгоритм Шора, має сенс розглянути і перспективні алгоритми ЕП, які у майбутньому можуть замінити нинішні. А саме, учасники що пройшли у другий раунд конкурсу NIST Post-Quantum crypto Project [12] такі, як: - Crystals Dilithium; - SPHINCS+ та SPHINCS; - XMSS.

1. Crystals Dilithium – алгоритм ЕП який базується на методі "Fiat-Shamir з переривання-ми", який використовує вибірку відхилення, для того, щоб зробити схеми Fiat-Shamir на основі решіток компактними та безпечними. Схема з найменшими розмірами підпису з використанням такого підходу – це схема Ducas, Durmus, Lepoint та Lyubashevsky, що базується на припущенні NTRU і вирішальним чином використовує гаусовську вибірку для створення підписів. Оскільки вибірку з Гауса важко здійснити безпечно та ефективно, вирішено використовувати лише рівномірний розподіл [13]. Dilithium покращує найефективнішу схему, яка використовує тільки рівномірний розподіл, завдяки Bai та Galbraith, використовуючи новий метод, який скорочує відкритий ключ більш ніж у 2 рази. Наскільки відомо, Dilithium має найменший відкритий ключ та розмір підпису у порівнянні з будь-якою схемою підпису на основі решітки, яка використовує лише рівномірну вибірку.

2. SPHINCS+ – оптимізований варіант SPHINCS криптографічної схеми, заснованої на криптографічній стійкості геш-функції. Вона включає в себе кілька поліпшень, спеціально спрямованих на зменшення розміру підпису. Схема включає в себе алгоритм формування та перевірки електронного цифрового підпису.

3. SPHINCS – представляє з себе високонадійну систему цифрового підпису [14], що заснована на геш-функції без збереження даних, яка представляє з себе безпеку «128 bit» проти атак квантового комп’ютеру (Табл. 7-8). Як свідкує аналіз, механізми ЕП класу SPHINCS, розроблені на достатньому рівні, при зрозумілих моделях загроз, в том числі за умов криптографічної стійкості до атак на основі стійкості до знаходження прообразу та стійкості до колізій. Оптимізаційні рішення SPHINCS полягають у тому, щоб включити у підпис усі вузли на певному рівні. Це дозволить уникнути зберігання автентифікаційних шляхів над тим рівнем. SPHINCS представляє дві нові ідеї, які суттєво зменшують розмір підпису.

По-перше, щоб підвищити рівень безпеки випадкового вибору індексу, SPHINCS замінює лист OTS схемою геш-підпису (FTS). FTS є, як випливає з назви, схемою підпису, яка призначена для підписання декількох повідомлень; в контексті SPHINCS це дозволяє кілька колізій індексів, що, в свою чергу, дозволяє зменшити висоту дерев для того ж рівня безпеки. Наприклад, SPHINCS-256 зменшує загальну висоту дерева з 256 до всього 60, зберігаючи при цьому захист від квантових нападників.

На відміну від SPHINCS, SPHINCS+ був кілька покращений. По-перше, був змінений захист від багатоцільових атак. При цьому, основна концепція полягає у використанні різної геш-функції для кожного виклику. Кожен виклик геш-функції виконується за допомогою іншого ключа і застосовує різні бітові маски. Ключі і бітові маски генеруються псевдовипадково з адреси, що вказує контекст виклику, і публічного нащадку. Щоб отримати абстракцію, були введені поняття налаштованих геш-функцій, які крім вхідного значення приймають адресу публічного нащадку.

Таблиця 7 – Розмір алгоритму SPHINKS

Підпис	41000 bytes
Розмір публічного ключа	1056 bytes
Розмір приватного ключа	1088 bytes

Таблиця 8 – Параметри алгоритму SPHINKS

Значення	Параметр
256	Довжина біта хешів у HORST і WOTS
512	Довжина біта хеша повідомлення
60	Висота гіпер-дерева
12	Шари гіпер-дерева
16	Параметр Winternitz, що використовується для підписів WOTS
2^{16}	Кількість секретних елементів HORST
32	Кількість виявлених елементів секретного ключа в кожному сигналі HORST

По-друге, стиснення відкритого ключа WOTS + відбувається з відкритим ключем: останні вузли ланцюжків WOTS + стискаються не за допомогою L-дерева, а за допомогою одного настроюваного виклику геш-функції. Цей виклик знову отримує адресу і загальнодоступне початкове значення для введення цього виклику і для генерації бітової маски протягом усього часу введення. Це було неможливо раніше, не підірвавши відкритий ключ. Тепер, коли бітові маски генеруються псевдовипадково, це не впливає на розмір відкритого ключа.

По-третє, HORST схеми з короткою підписом були замінені на FORS схеми. Перевага полягає в тому, що тепер є можливість використовувати набагато менші параметри і, тим самим, в кінцевому підсумку виграти в розмірі та швидкості підпису. Остання зміна стосується вибору індексу що перевіряється.

В SPHINCS пара ключів HORST, використовувана для підпису повідомлення, була обрана, генеруючи індекс псевдовипадковим чином. Оскільки задіяно секретне початкове значення, верифікатор не зміг перевірити, чи був цей індекс дійсно згенерований таким чином. Це мало недолік, що полягає в тому, що зловмисник, націлений на HORST, міг виконати багатоцільову атаку, використовуючи одне геш-обчислення для одночасного націлювання на всі екземпляри HORST. Це більше неможливо, оскільки кожне повідомлення, яке намагається згенерувати противник, тепер безпосередньо пов'язується з екземпляром FORS та стає непридатним для будь-якого іншого примірника. Крім того, це дозволяє опустити індекс в сигнатурі SPHINCS +. Ці зміни дозволили визначити набори параметрів з розмірами підпису.

4. XMSS – ЕП заснований на схемі підпису Меркла [15] і узагальненій схемі підпису Меркла (GMSS) [16]. XMSS є ефективною схемою пост-квантового підпису з мінімальними припущеннями безпеки. XMSS є ЕП подібним до SPHINCS, однак із більш коротшими підписами, за рахунок того, що вона фіксує дані. Наприклад, варіант XMSS-T генерує 8,8 кБ ЕП для продукування 2^{60} повідомлень та забезпечує 128-бітний квантовий захист. Головною відмінністю від SPHINCS є те, що гіпердерево SPHINCS ділиться на багато рівнів, так як ці дерева для кожного підпису мають в реальному часі повторно обчислюватися. Однак XMSS має пе-

ревагу за рахунок додаткового ефективного алгоритму, що дає змогу зекономити на вартості обчислення за рахунок створення багатьох підписів.

XMSS екзистенційно непідробний під прицільно обраних атак повідомлень в стандартній моделі. Схема XMSS дозволяє розширювати одноразові підписи [17]. Вимоги безпеки для XMSS мінімальні. Існування безпечної схеми підпису передбачає існування другого сімейства провізних стійких геш-функцій та сімейства псевдовипадкових функцій. XMSS є практичним, оскільки існує багато способів створити дуже ефективні (геш) сім'ї функцій, які, як вважають, є стійкими прообразами або псевдовипадковими, навіть за наявності квантових комп'ютерів. Наприклад, криптографічні геш-функції і блокові шифри можуть бути використані для цілей побудови таких сімей. Зокрема, існують такі конструкції, що базуються на складних задачах теорії алгебри та кодування. При цьому величезна кількість інстанцій XMSS гарантує довгострокову доступність надійних і ефективних схем підпису [18].

5 Висновки

В зв’язку з інтенсивним розвитком та впровадженням систем що реалізують децентралізований принцип управління, зросла потреба щодо забезпечення потрібних рівнів безпеки при використанні цих систем. В роботі розглянуті існуючі алгоритми захисту деяких криптографічних систем, зокрема алгоритми електронного підпису та функцій гешування, визначені мета і причина їх використання, вплив на відповідні системи, та на безпеку блокчейн в цілому. Наведені приклади найпоширеніших функцій гешування, а саме алгоритм SHA-256 та Keccak (він же SHA-3).

У роботі розглянута криптографія асиметричного шифрування, яка використовує відкриті та приватні ключі. Досліджено алгоритм ECDSA, який є алгоритмом з відкритим ключем для створення цифрового підпису, аналогічний за своєю побудовою до DSA. Крім того, проведено загальний аналіз та огляд перспективних електронних підписів і функцій гешування, які можуть замінити вже існуючі зразки. До таких слід віднести наступні:

- Crystals Dilithium – алгоритм, що базується на методі "Fiat-Shamir з перериваннями";
- SPHINCS+ – є оптимізованим варіантом криптографічної схеми SPHINCS;
- XMSS – ЕП, що заснований на схемі підпису Меркла.

У роботі розглядаються причини можливого використання даних алгоритмів у майбутньому, та аналізується їх стійкість перед «квантовими атаками», що є важливим аспектом існування та безпечного функціонування децентралізованих систем у майбутньому.

Посилання

[1] NISTIR 8202 Blockchain Technology Overview / Yaga D., Mell P., Roby N., Scarfone K. P. 50-59. URL: https://csrc.nist.gov/publications/detail/nistir/8202/final

[2] Mearian L. IBM Touts Quantum Computing Advance. URL: https://www.computerworld.com/article/2502275/ibm-touts-quantum-computing-advance.html

[3] Zalka Chr. Grover’s quantum searching algorithm is optimal. *Phys.Rev.*1999. A60. P. 2746 – 2751

[4] Shor P. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Foundations of Computer Science*. Proceedings 35th Annual Symposium on - IEEE. 1994. P.124 – 134.

[5] SHA-2. URL: https://ru.wikipedia.org/wiki/SHA-2

[6] Почему Keccak настолько крут и почему его выбрали в качестве нового SHA-3. URL: https://habr.com/ru/post/168707/

[7] Comparison of cryptographic hash functions. URL: https://en.m.wikipedia.org/wiki/Comparison_of_cryptographic_hash_ functions?wprov=sfti1

[8] NISTIR 7896. Third-Round Report of the SHA-3 Cryptographic Hash Algorithm Competition. *NIST*. 2012. P. 50 – 65.

[9] Digital Signature in Blockchain. URL: https://dzone.com/articles/digital-signature-2

[10] Johnson D., Menezes A., Vanstone S. The Elliptic Curve Digital Signature Algorithm (ECDSA). *International Journal of Information Security*. 2001. Vol. 1, Issue 1. P. 36 – 63.

[11] Haddaji R., Ouni R., Bouaziz S., Mtibaa A. Comparison of Digital Signature Algorithm and Authentication Schemes for H.264 Compressed Video (IJACSA). *International Journal of Advanced Computer Science and Applications*. 2016. Vol. 7, № 9. P.7.

[12] Post-Quantum Cryptography. URL: https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions

[13] CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation / Ducas L. and all. URL: https://pq-crystals.org/dilithium/data/dilithium-specification.pdf

[14] SPHINCS: practical stateless hash-based signatures/ Bernstein D.J. and all. URL: https://www.iacr.org/archive /eurocrypt2015/90560214/90560214.pdf

[15] Merkle R. A certified digital signature. *Advances in Cryptology - CRYPTO’ 89* : *Proceedings* /Gilles Brassard, ed. Vol. 435 of Lecture Notes in Computer Science. Berlin: Springer /Heidelberg, 1990. P. 218–238.

[16] Merkle signatures with virtually unlimited signature capacity/ Buchmann J. and all. *Applied Cryptography and Network Security*/ Katz J. and Yung M. ed.Vol. 4521 of Lecture Notes in Computer Science. Berlin: Springer / Heidelberg, 2007. P.31–45.
[17] Rogaway P., Shrimpton T. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. *FSE*/ Roy B.K., Meier W., ed. Vol. 3017 of Lecture Notes in Computer Science. Berlin: Springer / Heidelberg, 2004. P. 371–388.
[18] Buchmann J., Dahmen E., Hülsing A. XMSS - A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions. *Post-Quantum Cryptography. PQ Crypto 2011*. Vol. 7071 of Lecture Notes in Computer Science. Berlin: Springer / Heidelberg, 2011. P. 117–129.

Reviewer: Oleksandr Oksiiuk, Doctor of Sciences (Engineering), Full Professor, Taras Shevchenko National University of Kiev 81 Lomonosova St., Kyiv, 03189, Ukraine. E-mail: o.oksiuk@gmail.com

Received on March 2019.

Authors:
Oleksandr Yakishin, computer science student, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: dkrakaslian@gamil.com
Oleksandr Oniichichuk, computer science student, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: onik4524a@gmail.com
Volodymyr Skrynnik, Researcher, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: v.skrynnik@karazin.ua
Kateryna Kuznetsova, computer science student, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine.
E-mail: kate7smith12@gmail.com

Application of cryptographic algorithms in decentralized networks and prospects for their replacement for the post-quantum period.

Abstract. The article reviewed cryptographic algorithms that are used or can be used in decentralized networks. Algorithms for protection, both from classical and from quantum attacks are given. In the work there is a general analysis of the methods already used. Reviewed algorithms for protection, both from classical and from quantum attacks are given. The results of this work show the need for cryptographic algorithms to protect against possible quantum attacks in the future.

Keywords: Cryptographic algorithms; Algorithms for the protection of decentralized systems; Blockchain.

Рецензент: Александр Оксиюк, д.т.н., проф., Киевский национальный университет имени Т. Шевченко, ул. Ломоносова 81, Киев, 03189 Украина. E-mail: o.oksiuk@gmail.com

Поступила: Март 2019.

Авторы:
Александр Якишин, студент факультета компьютерных наук, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина. E-mail: dkrakaslian@gmail.com
Александр Оникийчук, студент факультета компьютерных наук, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина. E-mail: onik4524a@gmail.com
Владимир Скрынник, научный сотрудник, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина. E-mail: v.skrynnik@karazin.ua
Екатерина Кузнецова, студентка факультета компьютерных наук, Харьковский национальный университет имени В.Н. Каразина, Харьков, Украина. E-mail: kate7smith12@gmail.com

Применение криптоалгоритмов в децентрализованных сетях и перспективы их замены в постквантовый период.

Аннотация. В работе проведен обзор используемых в блокчейн системах электронных подписей и функций хеширования. Представлены криптографические алгоритмы, которые уже используются или могут использоваться в децентрализованных сетях. Приведены алгоритмы для защиты, как от классических, так и от квантовых атак. Результаты работы свидетельствуют о необходимости совершенствования криптографических алгоритмов для защиты от возможных квантовых атак в будущем.

Ключевые слова: криптографические алгоритмы; алгоритмы защиты децентрализованных систем; блокчейн.