УДК 004.056.5+681.3

# АСПЕКТИ ПРАКТИЧНОЇ РЕАЛІЗАЦІЇ КРИПТОВАЛЮТИ ETHEREUM

О. М. Богуцький[1, a]

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

**Анотація**

У даній статті досліджується та аналізується проблема стійкості криптовалюти Ethereum та можливість захисту від атак на реалізацію. Важливість цього питання саме для даної криптовалюти полягає у тому, що в якості протоколу узгодження застосовується будь-яке адаптивне правило реалізоване на віртуальній машині – Ethereum Virtual Machine. Переваги полягають у гнучкості, тобто будь-який алгоритм, будь-які операції, які необхідні клієнтам, можна реалізувати. Як наслідок, виникає багато лазівок, які складно передбачити та аналізувати.

*Ключові слова*: криптовалюта, Ethereum, Blockchain, децентралізована система

## Вступ

Ethereum – платформа для створення практично будь-яких децентралізованих онлайн-сервісів на базі блокчейна, що працюють на базі розумних контрактів. Реалізована як єдина децентралізована віртуальна машина. Це молода валюта, оскільки була запущена у літі 2015 року. Вона використовує власну віртуальну машину – Ethereum Vitrual Machine(EVM). Крім того, Ether – це програмований блокчейн, у який можна закласти будь-який алгоритм.

## 1. Структура Ethereum

### 1.1. Ланцюжок блоків

Ланцюжок блоків транзакцій (англ. *Blockchain*) – розподілена база даних, яка підтримує постійно зростаючий перелік записів, званих блоками, від підробки та переробки. Кожен блок містить часову мітку та посилання на попередній блок геш-дерева (дерева Меркле). Це своєрідний інструмент для збереження даних транзакцій. Розподілена база даних – це сукупність взаємопов'язаних баз даних, розподілених в одній комп'ютерній мережі. Логічний зв'язок між базами забезпечується системою керування, яка дозволяє керувати розподіленою базою даних так, щоб створювати у користувачів ілюзію цілісної бази даних.

За структурою Blockchain – ланцюжок блоків, який містить в собі певну інформацію. При цьому всі блоки ланцюжка пов'язані один з одним. Блок наповнений групою записів, а нові блоки завжди додаються в кінець ланцюга і дублюють інформацію, що міститься в раніше створених структурних одиницях системи, додаючи до неї нову. Цифрові записи об'єднуються у блоки, які пов'язуються в хронологічному порядку за допомогою криптографічних алгоритмів. Кожен блок пов'язаний із попереднім та містить у собі набір записів. Нові блоки додаються у кінець ланцюжка. Щоб транзакція вважалася підтвердженою, її формат і підписи повинні перевірити і потім групу транзакцій записати в спеціальну структуру – блок. Інформацію у блоках можна швидко перевірити. Кожен блок завжди містить інформацію про попередній.

Блок являє собою дерево геш-покажчиків або так зване «дерево Меркле» – структура даних, яка містить підсумкову інформацію про якийсь більший обсяг даних. Використовується для перевірки цілісності даних. Дані поділяються на малі частини, які індивідуально гешуються. Потім з кожної пари гешів по черзі обчислюється спільний геш. Якщо у гешу немає пари – він переноситься на новий рівень. Дана процедура повторюється доки не залишиться один геш.

Створений блок із доданим до нього деяким згенерованим випадковим числом буде прийнятий іншими користувачами, якщо числове значення гешу заголовка дорівнює або нижче певного числа, величина якого періодично коригується. Так як результат гешування непередбачуваний, немає алгоритму отримання цього випадкового числа, крім випадкового перебору. Дерева Меркле є зручним механізмом, оскільки володіють наступними важливими властивостями:

1) Proof-of-Existence. Властивість, що дає можливість перевірити наявність певних даних в операціях блокчейну. Тобто доведення того, що користувач справді здійснив усі необхідні обчислення для знаходження гешів транзакцій та цілого блоку. Доведення Меркле складається із елемента, кореневого гешу дерева та гілки, що включає усі геші від елемента до кореня. Завдяки цьому, можна легко переконатись, що було дотримано правил гешування протягом усієї вітки, що у свою чергу доводить, що елемент розміщений на правильному місці.

[a] sashabohutskiy@gmail.com

2) Заголовок будь-якого блоку знаходиться на основі гешів транзакцій і т. д. Тому, щоб обчислити геш заголовку, необхідно використовувати геші листків у дереві Меркле. Відповідно, при підробці чи спотворенні хоча б одного листка відбуватиметься непередбачувана зміна заголовку самого блоку.

Дерева Меркле – основа блокчейну Ethereum.

### 1.2. Доведення Меркле в Ethereum

Кожен блок в Ethereum містить не одне, а цілих три дерева Меркле для об'єктів різних типів[1]: транзакцій, квитанцій(тобто даних, про результати виконання кожної транзакції) та станів.

В Ethereum використовується префіксне дерево Меркле. Двійкові дерева Меркле гарні для перевірки інформації, представленої у вигляді списку. Вони непогано підходять і для подання дерев транзакцій, тому що редагувати їх не потрібно. Щодо дерева станів, то з ним все трохи складніше. В Ethereum стан – це таблиця ключів і значень, в якій ключі є адресами, а значення – рахунками з балансом, одноразовим кодом і сховищем (яке саме по собі є деревом).

Стан часто оновлюється: баланси і коди рахунків змінюються, користувачі створюють нові рахунки, ключі в сховище додаються і видаляються. Для роботи зі змінними даними бажано використовувати структуру, у якій можна легко обчислювати новий корінь після вставлення, оновлення, зміни або видалення даних, при цьому не обчислюючи заново все дерево. Також бажані дві інші властивості:

1) Глибина дерева повинна бути обмеженою через загрозу того, що зловмисник може спробувати створити якомога більш глибоке дерево для проведення DoS-атаки (щоб кожне оновлення дерева займало багато часу).
2) Корінь дерева повинен залежати тільки від даних, але не від порядку виконання оновлень. Застосування оновлень в іншому порядку і навіть перерахунок дерева з нуля не повинні змінювати корінь.

Префіксне дерево – це структура даних, яка, краще за все відповідає описаним критеріям. У такому дереві ключ, за яким зберігається значення, кодується у вигляді гілки дерева. У кожного вузла є 16 дочірніх вузлів, що дозволяє кодувати ключі в шістнадцятковій системі числення.

## 2. Атаки на реалізацію Ethereum: атака переповнення буферу

Атака переповнення буферу (англ. *Buffer Overflow*) – це явище, що виникає, коли комп'ютерна програма записує дані за межами виділеного їй буфера. Головна проблема таких атак полягає у тому, що зловмисник може поміщати в оперативну пам'ять інструкції на машинній мові та здійснювати будь-які необхідні йому дії. Найчастіше такі помилки виникають через недостатню перевірку вхідних даних. Атака переповнення буферу в криптосистемах є частковим випадком криптоаналітичних атак на реалізацію[3].

Правильно написані програми повинні перевіряти довжину вхідних даних для того, щоб переконатись, що вони не більші, ніж виділений буфер. Але на практиці цю вимогу виконати досить важко.

Переповнення буферу широко поширені серед програм, що написані на низькорівневих мовах програмування. Наприклад, асемблер чи C. У таких випадках програмісти повинні самостійно керувати розміром виділеної пам'яті[2]. А такі мови як Python, Java керують виділенням пам'яті самостійно, що робить помилки для переповнення буферу практично неможливими.

В Ethereum віртуальна машина контролює це на певному рівні, але не дає ніяких гарантій безпеки. Тому необхідно застосувати спеціальні методи захисту від атак на реалізацію. Особливо коли використовується гостьовий гіпервізор, то на його рівні можна прочитати та розібрати структуру віртуальної машини.

Розглянувши приклад[4], можна побачити, що у ньому не здійснюється ніяких перевірок на вхідні дані та атаки переповнення буферу можуть бути успішно реалізовані. Для забезпечення стійкості потрібно вводити додаткові перевірки на вхідні дані. Крім того, необхідні додаткові механізми у структурі EVM, що забезпечували б стійкість до таких атак.

## Висновки

У даному дослідженні вперше формулюється задача аналізу стійкості криптовалюти Ethereum до атак на реалізацію. Крім того, вперше досліджувалась стійкість до атаки «переповнення буферу». Оскільки раніше такі дослідження даної криптовалюти не проводились, то отримані результати можна використати для подальшого більш глибокого дослідження даної проблеми.

Провівши дослідження структури криптовалюти Ethereum з точки зору її стійкості до криптоаналітичних атак на реалізацію, можна зробити висновок про залежність успіху цієї атаки від вибору конкретного протоколу Ethereum. Оскільки у структурі EVM не має відповідних механізмів для захисту від такого роду атак, то для уникнення уразливостей необхідно здійснювати додаткову перевірку вхідних даних, хоч це і відіб'ється на швидкодії програми.

## Перелік використаних джерел


1. Ethereum Blog. Merkling in Ethereum. — Режим доступу: `https://blog.ethereum.org/2015/11/15/merkling-in-ethereum/`.
2. Джон Эриксон. Переполнение буфера. — 2010. — С. 139.
3. S. Ravi, A. Raghunathan, P. Kocher, S. Hattangady Security in Embedded Systems: Design Challenges ACM Transactions on Embedded Computing Systems, Vol. 3, No. 3. — August 2004. — P. 461-491.
4. Ethereum Project. Create a digital greeter. — URL: `https://www.ethereum.org/greeter`.