УДК 004

МОДЕЛИРОВАНИЕ ИЗМЕНЯЕМОГО БЛОКЧЕЙНА

Баскаков П. Е., Сафарьян О. А., Хабовец Ю. Ю.
Донской государственный технический университет, Ростов-на-Дону, Российская Федерация
baska.pavel@yandex.ru
safari_2006@mail.ru
yuriy131196@gmail.com

В работе поставлена задача — создать абстрактный изменяемый блокчейн на основе хеш-функций хамелеонов. Приведены общие сведения о теории использования хеш-функции с возможностью создания коллизий, показаны примеры листингов функций, необходимых для дальнейшей работы.

Ключевые слова: блокчейн, хеш, изменяемый блокчейн, хамелеон, криптовалюты, коллизия, алгоритм.

UDC 004

MODELLING OF REDACTABLE BLOCKCHAIN

Baskakov P.E., Safaryan O.A., Khabovets Yu.Yu.
Don State Technical University, Rostov-on-Don, Russian Federation
baska.pavel@yandex.ru
safari_2006@mail.ru
yuriy131196@gmail.com

The task of the paper is to create an abstract redactable blockchain based on a chameleon of hash functions. The paper provides general information about the theory of using hash functions with the ability to create collisions, shows examples of function listings required for further work.

Keywords: blockchain, hash, redactable blockchain, chameleon hash, cryptocurrency, collision, algorithm.

Введение. В наше время информационные технологии развиваются очень быстро. Поток информации стал настолько велик, что необходимость проверять достоверность и целостность полученных данных находится на одном из первых мест. Технология блокчейна, появление которой в 2009 году было связано с пиринговой платежной системой *Bitcoin*, на данный момент может быть распространена на любые взаимосвязанные информационные структуры и элементы. С помощью этой технологии осуществляется контроль за достоверностью информации, находящейся в цепочке блоков. Однако, как выяснилось, у блокчейна *Bitcoin* есть одна уязвимость — в него можно встроить контент, который никаким образом не относится к финансам [1].

Первые вставки нежелательного контента были зафиксированы в течение 2–3 лет с момента появления *Bitcoin*. Наличие в общедоступном блокчейне *Bitcoin* вставок подобного рода делает его автоматически незаконным на территории множества стран мира [2].

Идеи решения данной проблемы создателям *Bitcoin* пришли довольно быстро. Специальными группами разработчиков были созданы различные сервисы контроля транзакций. Через некоторое время было обнаружено, что хоть и количество встраиваемого контента уменьшилось, новая крайне нежелательная для блокчейна информация продолжала в нём появляться [3]. Помимо этого, процесс добавления новых блоков и неизменяемость блокчейна являются «бутылочным горлышком» на пути создания новых приложений и перехода к *Bitcoin* 2.0 [2].

В 2016 г. произошло событие, которое может изменить криптовалюты — консалтинговая компания *Accenture* подала заявку для получения патента на технологию изменяемого блокчейна, в основе которой лежит применение особых хэш-функций, называемых хамелеонами. В 2017 г. патент был успешно получен. На данный момент о готовности продукта, способного обеспечить изменяемость блокчейна, известно лишь то, что он находится на стадии прототипа. Представители компании *Accenture* заявили, что разрабатываемая платформа будет иметь открытый исходный код [4].

Постановка задачи. Задачей данного исследования является создание абстрактной модели изменяемого блокчейна некоторой криптовалюты на основе применения хеш-функций хамелеонов.

Теоретические сведения. Основу изменяемости блокчейна составляет применение функций хамелеонов. Они представляют собой криптографическую хеш-функцию, обладающую отличительным свойством — наличием специального ключа-«лазейки», использование которого позволяет генерировать коллизии. Идея хеш-хамелеонов впервые была представлена учеными Хьюго Кравчиком и Тал Рабин на симпозиуме *NDSS* (*The Network and Distributed System Security*) в 2000 г.

Вначале обобщим стандартную концепцию хэш-хамелеонов, чтобы сделать ее более актуальной на практике. Хэш-хамелеон можно рассматривать как кортеж эффективных алгоритмов:

$$CH = (HGen, Hash, HVer, HCol),$$

где каждый из них возвращает следующие значения:

- $(hk, tk) \leftarrow \$\, HGen(1^k)$ — алгоритм генерации вероятностного ключа *HGen* принимает за входной параметр безопасности $k \in N$ и выводит открытый хэш-ключ *hk* и секретный ключ-ловушку *tk*;
- $(h, r) \leftarrow \$\, Hash(hk, m)$ — вероятностный алгоритм хэширования *hk* и сообщения $m \in M$ возвращает пару (*h*, *r*), где $r \in Rhash$ обозначает случайные монеты, используемые для генерации хэша;
- $d = HVer(hk, m, (h, r))$ — алгоритм детерминированной проверки $HVer$ принимает в качестве входных данных сообщение $m \in M$ и пару (h, r) и возвращает 1 тогда и только тогда, когда $Hash(hk, m; r) = h$. В противном случае возвращает 0;
- $\xi' \leftarrow \$\, HCol(tk, (h, m, \xi), m')$ — вероятностный алгоритм поиска коллизий $HCol$ принимает на вход ключ-лазейку tk, допустимый кортеж (h, m, ξ) и новое сообщение $m' \in M$ и возвращает новую контрольную строку ξ', такую, что

$$HVer(hk, m, (h, \xi)) = HVer(hk, m', (h, \xi')) = 1.$$

Если (h, ξ) не является допустимой парой хэша/проверки для сообщения m, алгоритм возвращает $\perp$.

Опишем пример реализации хеш-функции, которая будет использоваться в данной работе.

Пусть p, q — простые числа, причем $p = 2q + 1$, и g — генератор подгруппы квадратичных вычетов QR_p в Z_p^*. Определим следующую хеш-функцию $(HGen, Hash, HCol)$:

- $(y, x) \leftarrow \$\, HGen(1^k)$, где x — ключ-ловушка tk. Он представляет собой случайное значение из промежутка $x \in [1, q - 1]$, а y — ключ хеша hk, который связан с генератором g следующим равенством:

$$y = g^x.$$

- $h := Hash(y, m; r, s)$. Чтобы захешировать сообщение $m \in \{0,1\}^*$, определим случайные $r, s \leftarrow \$\, Z_q$, и вернем

$$h := r - (y^{H(m||r)} \cdot g^s \bmod p) \bmod q,$$

где $H : \{0,1\}^* \rightarrow Z_q$ — стандартная устойчивая к коллизиям хеш-функция.

- $(r', s') \leftarrow \$\, HCol(x, (h, m, r, s), m')$ Чтобы найти коллизию для сообщения m', возьмем случайное число $k \in [1, q - 1]$ и вычислим:

$$r' := h + (g^k \bmod p) \bmod q,$$
$$s' := k - H(m'||r') \cdot x \bmod q.$$

Пара (r', s'), состоящая из случайного числа и нового значения хеша является результатом

работы алгоритма нахождения коллизии [4].

Описание программного средства. Разрабатываемое программное средство будет выполнять следующие функции:

- автоматическое моделирование структуры блокчейна,
- изменение существующих блоков,
- генерация коллизий между блоками с помощью хеш-хамелеонов,
- вывод информации, содержащейся в блоке.

На данный момент реализованы все основные элементы, необходимые для работы хеш-хамелеонов. Основная проблема в данной задаче — это очень большие числа. Так, чтобы посчитать хеш для одного сообщения, потребуются вычислительные мощности, сравнимые с почти десятком среднестатистических компьютеров.

Ниже на рис. 1–3 приведены фрагменты кода основных функций, принадлежащих алгоритму нахождения хеш-хамелеона.

```
def Hgen():
    x = random.randint(1, q - 1)
    y = pow(g, x, q)

    return x, y

```

Рис. 1. Листинг функции генерации *x* и *y*

```
def Hash(m: int, y):
    r = random.randint(0, q)
    s = random.randint(0, q)

    m_bytes, r_bytes = map(lambda num: num.to_bytes(num.bit_length() // 8 + 1, 'big').lstrip(b'\x00'), (m, r))

    sha = int(hashlib.sha1(m_bytes + r_bytes).hexdigest(), 16)

    h = r - (pow(y, sha) * pow(g, s, p)) % q

    return h
```

Рис 2. Листинг функции хеширования сообщения *m*

```
def Hcol(x: int, _m: int, h: int):
    k = random.randint(0, q - 1)
    _r = h + pow(g, k, p) % q

    _m_bytes, _r_bytes = map(lambda num: num.to_bytes(num.bit_length() // 8 + 1, 'big').lstrip(b'\x00'), (_m, _r))

    sha = int(hashlib.sha256(_m_bytes + _r_bytes).hexdigest(), 16)

    _s = k - (sha * x) % q

    return _r, _s
```

Рис. 3. Листинг функции генерации коллизии двух блоков блокчейна

Пример работы хеш-функции продемонстрирован на рис. 4. От исходного сообщения считается значение хеша, затем для сообщения генерируется коллизия и происходит валидация путем получения хеша от коллизии. Как видно из рисунка, хеши от исходного сообщения и от коллизии совпали, а значит коллизия подобрана верно.

```
Сообщение: Добрый вечер
Хеш от сообщения: 318060f569ed946577c8404a9e9c8b9bad6f34beca38fecd806761c6aa4d6768
Коллизия: nsQPAhwYfqLt
Хеш от коллизии: 318060f569ed946577c8404a9e9c8b9bad6f34beca38fecd806761c6aa4d6768
```

Рис. 4. Пример работы хеш-функции

Заключение. В настоящее время ведется работа над программным средством по моделированию изменяемого блокчейна. Ключевая функция хеш-хамелеон была успешно реализована. В дальнейшем будут реализованы функции для генерирования структуры блокчейна и перезаписи (удаления) транзакций внутри блоков.

Библиографический список

1. Juha Partala. Provably Secure Covert Communication on Blockchain. Physiological Signal Analysis Team, Center for Machine Vision and Signal Analysis, University of Oulu, P.O. Box 5000, FI-90014 Oulu, Finland, 2018 – 18 с.

2. Roman Matzutt, Martin Henze, Jan Henrik Ziegeldorf, Jens Hiller, Klaus Wehrle. Thwarting Unwanted Blockchain Content Insertion. Communication and Distributed Systems RWTH Aachen University Aachen, Germany, 2018 – 8 с.

3. Roman Matzutt, Jens Hiller, Martin Henze, Jan Henrik Ziegeldorf, Dirk Mullmann, Oliver Hohlfeld, and Klaus Wehrle. A Quantitative Analysis of the Impact of Arbitrary Blockchain Content on Bitcoin. 22nd International Conference on Financial Cryptography and Data Security 2018 – 18 с.

4. Giuseppe Ateniese, Bernardo Magri, Daniele Venturi, Ewerton Andrade. Redactable Blockchain or Rewriting History in Bitcoin and Friends. 2017 – 38 с.