УДК 004.056.5

П.И. СТЕЦЕНКО, Г.З. ХАЛИМОВ, д-р техн. наук

ПРОБЛЕМА СОВЕРШЕНИЯ УСЛОВНЫХ ТРАНЗАКЦИЙ В ЗАКРЫТЫХ BLOCKCHAIN-СИСТЕМАХ

Достижение консенсуса в открытых Blockchain-системах (например, Bitcoin и Ethereum), где отсутствует доверие между пользователями и имеются злоумышленники, является актуальным [1, 2]. Достижение консенсуса в распределенной системе, включающей злоумышленника, будет невозможным при условии, что сообщения не могут быть доставлены в течение ограниченного времени [3]. Базовые механизмы достижения консенсуса никогда не достигают консенсуса в случае умышленных задержек распространения сообщений [4].

Использование технологии Blockchain в закрытой, контролируемой среде в последнее время становится востребованным среди многих компаний. В 2015 г. одиннадцать банков успешно сотрудничали в развертывании закрытого регистра Blockchain проекта Ethereum для обработки транзакций [5].

В данной работе рассмотрена проблема, которая лишает пользователя возможности совершать транзакции на основе текущего состояния регистра Blockchain, что может привести к потере цифровых активов или атакам двойной траты. Показано, что некоторые смарт-контракты также могут иметь данную проблему. Представленные в работе результаты описывают риск использования технологии Blockchain в закрытой, контролируемой среде без учета ее сложных конструктивных особенностей.

1. Анализ связанных работ

Сравнительный анализ механизмов достижения консенсуса на основе доказательства проделанной работы (PoW) и протоколов с византийской отказоустойчивостью (BFT) проводился в работах [6 – 8]. BFT обеспечивает более низкую задержку и более высокую пропускную способность при обслуживании транзакций, чем PoW-механизмы. Недостатками BFT-протокола достижения консенсуса являются ограничения масштабируемости и вследствие этого необходимость использования сегментирования.

Вероятностные гарантии PoW-механизма достижения консенсуса и гарантии детерминирования BFT-протоколов исследуются в работе [8]. Достижение консенсуса на основе PoW сопоставляется с BFT-протоколами по двум ключевым моментам – масштабируемости и производительности. В результате исследования установлено, что PoW-протоколы считаются масштабируемыми, но неэффективными; а BFT-протоколы, напротив, считаются эффективными, но не масштабируемыми.

Одним из проектов, позволяющих избежать рассматриваемую в данной работе проблему, является PeerCensus [9]. Этот алгоритм состоит из двух компонентов: 1) выполнение BFT-протокола поверх Bitcoin с простой системой голосования; 2) сведение к минимуму последствий атак Сибиллы во время таких голосований. Последний компонент мешает злоумышленнику создавать множественные идентификаторы, чтобы превзойти число голосов своими собственными голосами.

Определение 1. Атака Сибиллы – разновидность атаки двойной траты, в которой злоумышленник наполняет сеть подконтрольными ему узлами, вследствие чего остальные пользователи смогут подключиться только к блокам, созданным злоумышленником.

Используя такой подход, PeerCensus усиливает гарантии Bitcoin и немедленно устраняет разветвления, что позволяет избежать проблемы совершения условных транзакций.

Определение 2. Условная транзакция – это транзакция, которая должна выполняться только в текущем наблюдаемом фиксированном состоянии регистра Blockchain или в более позднем его состоянии.

Аномалия протокола Paxos, схожая с рассматриваемой в данной работе проблемой, анализировалась в работах [10]. Следует отметить, что аномалия протокола Paxos отличается от

проблемы совершения условных транзакций, так как может произойти по двум транзакциям, выпущенным одним и тем же пользователем. В Blockchain-системах для упорядочения транзакций используется метка времени. Другим отличием является то, что, если консенсус достигнут, индекс решения не может измениться, а проблема совершения условных транзакций строго обусловлена тем, что индекс определенной транзакции или порядок блока данной транзакции в регистре Blockchain может измениться. Однако для приложений, которым не нужны параллельные зависимые запросы, достаточно протокола Paxos [10].

2. Основные сведения

Большинство Blockchain-систем отслеживают транзакцию путем включения ее в блок. Это осуществляется с помощью процесса майнинга перед добавлением к цепочке существующих блоков, которая и является регистром Blockchain. Механизм достижения консенсуса гарантирует общий порядок этих блоков, так что цепочка блоков в конечном итоге не становится деревом. В процессе функционирования системы возможен вариант, когда несколько новых блоков будут одновременно присоединены к последнему блоку регистра – такой временный процесс известен как разветвление. Как только обнаруживается разветвление, это означает, что участники знают о двух альтернативных версиях регистра Blockchain (ветвях), и самая длинная ветвь принимается как действительная. Blockchain-системы обычно предполагают, что разветвления могут вырасти до некоторой ограниченной глубины, так как рост ветви требует решения сложной вычислительной проблемы, которая сводится к трате большого количества времени, в течение которого кто-то получит информацию о самой длинной ветви регистра Blockchain. Криптовалюта Bitcoin предполагает присоединение еще шести блоков к регистру после выпущенной транзакции, чтобы данная транзакция рассматривалась как принятая системой. Аналогично проект Ethereum подразумевает присоединение от пяти до одиннадцати блоков после выпущенной транзакции для принятия ее системой [2].

Однако консенсус не может быть достигнут, если нет верхнего предела времени для доставки сообщения и если какой-то участник может потерпеть неудачу [3]. Консенсус обычно выражается тремя свойствами:

1) соглашение – если два корректных участника принимают решение, то они принимают его на одном и том же блоке;

2) корректность – блок, на котором принималось решение, должен быть одним из блоков, которые были предложены участниками;

3) завершение – в конечном итоге правильный участник принимает решение.

Процесс принятия общего решения в протоколах Paxos и Raft разработан таким образом, чтобы гарантировать свойства соглашения и корректности при получении сообщений с задержкой [11, 12]. Это достигается за счет того, что алгоритм жертвует свойством завершения, чтобы гарантировать, что только корректные ответы, удовлетворяющие свойствам корректности и соглашения, могут быть возвращены. Эти алгоритмы консенсуса являются привлекательными, потому что, если через некоторое время сеть стабилизируется и сообщения будут доставляться в ограниченное время, то консенсус будет достигнут [4].

В качестве базовой системы для проведения анализа выбран проект Ethereum, так как это основная Blockchain-система, позволяющая развертывание закрытых регистров.

2.1. Blockchain системы

Регистр Blockchain можно рассматривать как реплицируемую машину состояний, в которой обратная связь между блоками является указателем от текущего состояния к его предыдущему состоянию (рис. 1) [13].

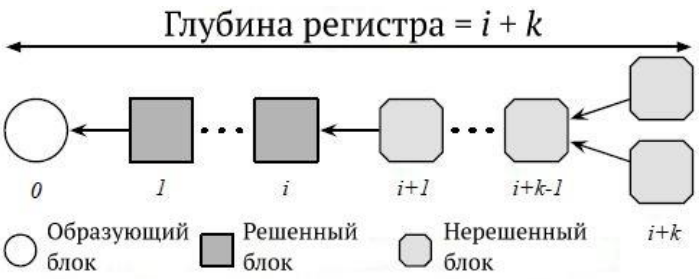

Рис. 1. Структура регистра Blockchain

Структура регистра Blockchain начинается с образующего блока с индексом 0 и связывает последовательно блоки в порядке, обратном их индексу; новый решенный блок имеет индекс $i > 0$, когда глубина регистра достигает $i + k$. Регистр, имеющий глубину 0, – это образующий блок.

Консенсус необходим для того, чтобы полностью упорядочить блоки, сохранив структуру цепочки блоков. В традиционных Blockchain-системах применяется PoW-механизм достижения консенсуса, требующий затрат на решение сложной вычислительной задачи [14]. Данный метод используется для того, чтобы достичь консенсуса, несмотря на произвольные неудачи, в том числе злонамеренное поведение пользователя. Специализированные одноранговые узлы (майнеры) должны доказуемо решить криптографическую HashCash задачу, прежде чем новый блок может быть добавлен к регистру Blockchain [15].

PoW-механизм достижения консенсуса находится в основе децентрализованной криптовалюты Bitcoin [1]. Примечательно, что PoW-консенсус не гарантирует достижения консенсуса детерминировано. Вместо этого он гарантирует, что консенсус будет достигнут с некоторой вероятностью, близкой к 1. Сложность решения криптографических головоломок, используемых в криптовалюте Bitcoin, приводит к присоединению нового блока к регистру Blockchain каждые 10 минут. Преимуществом этого длительного периода является то, что разветвления на регистре Blockchain из-за одновременного присоединения новых блоков возникают достаточно редко и Bitcoin решает эти разветвления, выбирая самую длинную ветвь и отбрасывая другую (другие). Проект Ethereum – это современная криптовалютная платформа с открытым исходным кодом, которая также опирается на PoW-консенсус [16]. В противоположность протоколу достижения консенсуса Bitcoin Ethereum генерирует один блок каждые 12-15 секунд. В то время как это увеличивает пропускную способность (количество транзакций в секунду), это также способствует увеличению количества переходных разветвлений регистра, так как вероятность одновременного присоединения майнерами новых блоков к регистру существенно возрастает. Чтобы избежать частой напрасной траты ресурсов на осуществление процесса майнинга, для разрешения разветвлений Ethereum использует протокол GHOST (Greedy Heaviest Observed Subtree), который не обязательно отбрасывает все блоки, не принадлежащие основной ветви. Проект Ethereum предлагает Тьюринг-полный язык программирования, который может быть использован для написания смарт-контрактов, определяющих новые правила владения активами [16].

2.2. Свойство завершения механизма достижения консенсуса

Путем ослабления свойства соглашения механизма достижения консенсуса Blockchain-системы могут гарантировать завершение достижения консенсуса детерминированно. В контексте Blockchain завершение механизма достижения консенсуса указывает на то, что блок был решен для следующего доступного индекса.

Пусть все транзакции решенного блока совершены. Здесь используется термин "совершенный", а не "подтвержденный", так как, в Blockchain терминологии транзакция предполагается "подтвержденной" иногда, когда только над ее блоком осуществлен процесс майнинга, а иногда, когда над $k + 1$ блоками осуществлен процесс майнинга (собственно блок транзакции и k последующих блоков).

Такое решение на основе включения блока в цепочку необходимо для обменных платформ криптовалют. Например, чтобы определить, что монеты того или иного вида, которые были недавно созданы (не добавлены посредством майнинга) в пределах этого блока, могут быть конвертированы в монеты криптовалют различного типа или в декретные валюты (например, EUR, USD). В частности, обзор того, что над блоком был осуществлен процесс майнинга и этот блок добавлен к цепочке блоков, не является достаточным условием для гарантирования решения: этот блок может быть частью одной переходной ветви без достигнутого (пока что) консенсуса ни по одной из этих переходных ветвей.

На рис. 1 представлено прекращение механизма достижения консенсуса на индексе *i* регистра Blockchain. Стрелка, направленная слева направо, указывает на то, что блок содержит хэш своего блока-предшественника, который расположен сразу с левой стороны от него. Блоки, над которыми только что осуществлен процесс майнинга, добавляются в правый конец регистра Blockchain, что может привести к кратковременному разветвлению, в случае если над несколькими блоками, относящимися к одному и тому же предшественнику, процесс майнинга совершен одновременно. Разветвления являются лишь переходными и их решение зависит от используемой Blockchain-системы. Механизм достижения консенсуса для индекса *i* заканчивается, когда участники принимают решение о присвоении новому блоку индекса *i*. Решение на блоке с индексом *i* имеет место для всех $i > 0$, когда глубина регистра Blockchain достигает $i + k$, где $k \geq 0$ является константой, зависящей от используемой Blockchain-системы.

Различные Blockchain-системы могут принимать различные значения *k*, чтобы определять завершение механизма достижения консенсуса. В Bitcoin (btc), $k_{btc} = 5$, что означает, что блок с индексом *i* решен (т.е., достижение консенсуса для индекса *i* завершается), когда над $k_{btc} + 1 = 6$ блоками с индексами *i*, …, *i* + 5 был успешно проведен процесс майнинга. Как уже упоминалось ранее, в криптовалюте Bitcoin новый блок решается каждые 10 минут. Следовательно, время совершения транзакции в Bitcoin $k_{btc} + 1 \ *10мин = 1час$. В проекте Ethereum (eth), начиная с версии 1.3.5 Homestead, $k_{eth} = 11$, что означает, что блок с индексом *i* решен (т.е., достижение консенсуса в отношении индекса *i* завершается), когда глубина регистра Blockchain достигает *i* + 11. Поэтому время для совершения транзакций в Ethereum равняется $k_{btc} + 1 \ *15сек = 180сек$. Следует обратить внимание, что некоторые обменные платформы криптовалют принимают разные значения *k* для регулирования вероятности соглашения. Например, обменная платформа QuadrigaCX ждет, пока над $k'_{btc} + 1 = 4$ блоками будет осуществлен процесс майнинга в регистре Blockchain криптовалюты Bitcoin, в то время как для регистра Ethereum данная платформа ждет осуществление процесса майнинга над $k'_{eth} + 1 = 12$ блоками [17].

3. Проблема совершения условных транзакций в закрытых Blockchain-системах

В данной работе рассмотрена проблема совершения условных транзакций, которая затрагивает ведущие Blockchain-системы, чей протокол достижения консенсуса не является гарантированно детерминированным.

Причины возникновения проблемы совершения условных транзакций. Данная проблема связана с асинхронностью сети, в которой задержки сообщения не могут быть ограничены, и со свойством завершения механизма достижения консенсуса.

Пусть два майнера осуществляют процесс майнинга на общей цепи блоков, начиная с общего образующего блока. Достаточно длительная задержка в сообщениях между ними может привести к тому, что майнеры будут, казалось бы, иметь согласие по состоянию регистра отдельно на разных ветвях цепи, содержащих более *k* блоков каждая, для любого *k*.

Данное обстоятельство является серьезной проблемой, так как может привести к простым атакам внутри любой закрытой, частной сети, в которой пользователи имеют стимул для максимизации своих прибылей – в виде монет, фондовых опционов или произвольной

собственности. Кроме того, этот сценарий является реалистичным в контексте частной сети, где сотрудники компании имеют прямой доступ к некоторым сетевым ресурсам. Когда сообщения в итоге доставляются, результаты несогласованности создают противоречия на регистре Blockchain.

Неподтвержденные транзакции. Проблема совершения условных транзакций, в которой транзакция t_i становится подтвержденной в рамках слота i с точки зрения некоторых узлов, представлена на рис. 2.

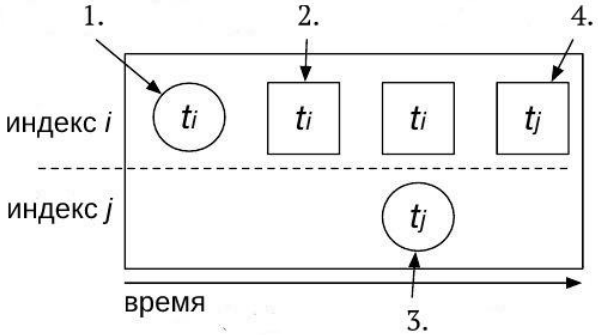

Рис. 2. Проблема совершения условных транзакций

На рис. 2:

1. Транзакция t_i предложена.
2. Транзакция t_i кажется подтвержденной.
3. Транзакция t_j предложена другим узлом.
4. Транзакция t_j подтверждена первой.

Основываясь на данном наблюдении, один пользователь предлагает новую транзакцию t_j, зная, что t_i успешно подтверждена. Можно представить простой сценарий, в котором "Боб передает некоторую сумму средств Алексу" (t_j) только тогда, когда ранее "Боб успешно получил деньги от Алисы" (t_i). Однако, как только эти узлы будут уведомлены о другой ветви подтвержденных транзакций, они примут решение реорганизовать ветвь для решения разветвления. Реорганизация удаляет подтвержденную транзакцию t_i из слота i. Позже транзакция t_j будет успешно подтверждена в слоте i.

Первый пользователь выпускает транзакцию t_i, над которой успешно проводится процесс майнинга и которая подтверждается. Далее второй пользователь выпускает транзакцию t_j, которая обуславливается подтверждением t_i. Следует отметить, что должно выполняться $j \geq i + k$ для того, чтобы t_i была подтверждена, перед тем как t_j будет выпущена. Однако транзакция t_j в итоге реорганизовывается и успешно подтверждается перед t_i, тем самым нарушая зависимость между транзакциями t_i и t_j.

Проблема, рассматриваемая в данной работе, связана с нарушением зависимости между транзакциями t_j и t_i: Транзакция t_j произошла и это значит, что Боб передал денежную сумму Алексу, однако транзакция t_i не произошла, а это значит, что Боб не получал деньги от Алисы. Следует обратить внимание, что в криптовалюте Bitcoin транзакция t_i будет отброшена, тогда как в Ethereum транзакции t_i в некоторых случаях могут быть подтверждены в слоте j.

Упрощение реализации атаки двойной траты. Одним из очевидных и серьезных последствий проблемы совершения условных транзакций является возможность злоумышленника провести атаку двойной траты: например, дважды конвертируя все свои монеты в товары. Сценарий состоит из выпуска злоумышленником первой транзакции t_1, которая конвертирует все его монеты на товары в блоке i, и старта майнинга блоков после того, как блок $i-1$ находится в изоляции сети. Как часть этого процесса майнинга злоумышленник осуществляет процесс майнинга над другой транзакцией t_2, при помощи которой он также конвертирует все свои монеты в товары. Затем злоумышленник ждет, пока глубина регистра Blockchain не достигнет $i+k$, после чего может забрать свои товары как результат транзакции t_1. Далее злоумышленник публикует свою более длинную цепочку, не содержащую транзакции t_1, таким образом, чтобы эта цепочка была принята остальной сетью. Транзакция t_2 под-

тверждается в блоке *j*, и, после того как глубина цепочки блоков достигнет $j+k$, злоумышленник может забрать свои товары во второй раз. Следует обратить внимание, что даже если кто-то пытается позже вновь подтвердить транзакцию t_1, то сделка будет считаться недействительной, так как средств на балансе окажется недостаточно.

Отслеживание проблемы совершения условных транзакций. Другим серьезным аспектом рассматриваемой проблемы является незаметное выполнение. Точнее, данная проблема полагается на ошибочно подтвержденное состояние регистра Blockchain. После того, как ошибочно подтвержденное состояние становится неподтвержденным, нет никакого способа рассмотреть это проблемное состояние цепочки блоков и заметить проблему совершения условных транзакций. Несмотря на то, что можно наблюдать, как одноранговый узел осуществляет процесс майнинга над несколькими блоками подряд, отсутствует какой-либо способ отследить бенефициаров данной проблемы. Таким образом, это подталкивает пользователей закрытых регистров эффективно использовать рассматриваемую проблему для атаки на регистр Blockchain.

4. Смарт-контракты как способ решения проблемы совершения условных транзакций

Смарт-контракты являются основополагающим аспектом системы Ethereum. Смарт-контракты могут быть запрограммированы обеспечить выполнение кода при определенных условиях. Установлено, что предотвращение возникновения проблемы совершения условных транзакций тесно связано с корректным программированием смарт-контракта. Это означает, что если смарт-контракт был запрограммирован так, что соответствующим образом не проверено условие появления первой транзакции, то будет выполняться, действуя как нормальная, транзакция.

Вычисление на регистре и вне регистра Blockchain. Смарт-контракт conditionalPayment, написанный на языке Solidity, при котором не возникает проблемы совершения условных транзакций, представлен ниже.

```
1 contract conditionalPayment {
2
3 uint32 paid; // для отслеживания за выплаченной Алисой суммой при принятии решения о переводе средств Боба
4 mapping (address => uint256) public balances; // сопоставление адресов с соответствующим им балансом
5 address A = 0x57ec7927841e2d25aad5f335e3b701369b177392; // адрес счета Алисы
6 address B = 0x5ae58375c89896b09045de349289af9034902905; // адрес счета Боба
7
8 modifier onlyFrom(address _address) { // позволяет выполнение функций в зависимости от вызывающего
9 if (msg.sender != _address) throw;
10 _
11 }
12
13 function sendTo(address B, uint32 _amount) onlyFrom(A) { // Алиса отправляет деньги Бобу
14 if (balances[A] >= _amount) { // проверка достаточно ли доступных средств на счете
15 balances[A] -= _amount;
16 balances[B] += _amount;
17 paid = _amount; // сортировка выплачиваемой суммы
18 }
19 }
20
21 function sendIfReceived(address C, uint32 _amount) onlyFrom(B) { // Боб отправляет деньги Алексу
22 if (paid > _amount) { // только если предыдущий платеж был успешным
23 balances[B] -= _amount;
24 balances[C] += _amount;
25 } else {
```

```
26 throw; // отмена выполнения контракта
27 }
28 }
29 }
```

Ключевой особенностью является то, что функция sendIfReceived группирует два шага: проверяет, была ли сумма оплачена на строке 22, и платеж, полученный в результате этой успешной проверки на строках 23 и 24. Поскольку эти два шага выполняются на цепочке блоков, то для того, чтобы второе условие произошло, первое должно быть обязательно истинным.

Однако, если эти два шага были частью двух отдельных функций контракта, одна из них, проверяющая, что сумма была уплачена, и другая, которая будет производить платеж и будет использовать возвращенную стоимость первой, в данном случае может возникнуть проблема совершения условных транзакций.

Рассмотрим функцию checkPayment в смарт-контракте problematicConditionalPayment, представленном ниже.

```
1 contract problematicConditionalPayment {
2 ...
3 function checkPayment(address B, uint32 _amount) onlyFrom(B) constant returns
(bool result) {
4 if (paid > _amount) { // проверка, что Алиса заплатила
5 return true;
6 } else throw;
7 }
8
9 function sendIfReceived(address C, uint32 _amount) onlyFrom(B) { // Боб от-
правляет деньги Алексу
10 balances[B] -= _amount;
11 balances[C] += _amount;
12 }
13 }
```

Данная функция проверяет, что платеж от Алисы прошел корректно (строки 3 – 7), а другая функция sendIfReceived была изменена для выполнения платежа без каких-либо условий (строки 9 – 13).

Даже если Боб вызывает функцию checkPayment и замечает, что функция выполняется успешно до вызова sendIfReceived, может возникнуть рассматриваемая в работе проблема. Причина в том, что проверка сделана вне регистра Blockchain, и ничто не гарантирует, что платеж от Алисы не был реорганизован, в то время как Боб проверял результат в автономном режиме.

Смарт-контракт conditionalPayment имеет более высокие шансы не быть подверженным проблеме совершения условных транзакций, поскольку выполняет на регистре Blockchain как проверку, так и перевод средств после выполнения условия. Однако это не гарантирует, что такой смарт-контракт застрахован от рассматриваемой проблемы. Для формального обоснования данного утверждения необходимо дальнейшее исследование. Кроме того, как и транзакции, смарт-контракты должны быть включены в блок, над которым осуществляется процесс майнинга и который добавляется к цепочке блоков. Включение его в регистр Blockchain, даже с k последующими блоками, может также оставаться подверженным переупорядочению и приводить к другим типам проблем совершения условных транзакций.

Мультиподпись и вариант криптовалюты Bitcoin. Даже без Тьюринг-полного языка программирования в криптовалюте Bitcoin могут быть способы обойти проблему совершения условных транзакций. Идея заключается в том, чтобы изменить условную транзакцию на совместный платеж, который включает в себя как условную транзакцию, так и действие, обеспечивающее это условие. Идея включения транзакции и действия аналогична идее группировки в функции SendIfReceived смарт-контракта conditionalPayment, проверки и перевода средств, которые описаны ранее.

Совместный платеж будет представлять собой совершение платежа Алексу как Алисой, так и Бобом. Таким образом, платеж будет принимать два входа, принадлежащие разным

людям и дающие один выход. Поскольку монеты этих двух входов поступают с разных адресов, то для совместного платежа необходимы две разные подписи. Совместный платеж может быть достигнут при помощи транзакции с использованием мультиподписи в криптовалюте Bitcoin. Такая транзакция для выполнения требует две подписи от Алисы, Боба и стороннего арбитра. Если и Алиса, и Боб подписывают транзакцию, то она выполняется, и Алекс получает деньги. Однако, если Алиса или Боб отказываются подписывать транзакцию, то арбитр может помочь в разрешении транзакции путем своей подписи. Важно отметить, что семантика совместного платежа отличается от условной транзакции тем, что Боб не может ждать, пока получит деньги от Алисы, чтобы выбрать, что делать вне зависимости от платежа Алексу.

5. Обсуждение

Следует отметить, что в PoW-механизме достижения консенсуса имеется одна важная деталь. Если система достаточно велика и вычислительная мощность майнинга достаточно рассеяна среди достаточного количества пулов майнинга, то вероятность наличия пула майнинга, осуществляющего процесс майнинга быстрее других, может быть сделана сколь угодно малой. По этой причине рассматриваемая в работе проблема совершения условных транзакций имеет очень низкую вероятность появления в крупномасштабных свободно доступных Blockchain-системах, таких как Bitcoin. Для майнеров существуют стимулы не раскрывать блок, над которым они успешно осуществили процесс майнинга, чтобы другие тратили усилия на майнинг. Это, в свою очередь, позволяет им присоединить более длинную цепочку блоков, чем другим, а это может потенциально привести к проблеме совершения условных транзакций [18].

Атака 51 процента может привести к проблеме совершения условных транзакций в открытой Blockchain-системе.

Определение 3. Атака 51 процента – атака, в которой злоумышленник, контролирующий более половины вычислительной мощности майнинга в общедоступной сети, может быстрее осуществлять процесс майнинга над блоками по сравнению с другими майнерами.

Злоумышленник может выпустить транзакцию для конвертации некоторого количества монет Bitcoin, чтобы снять некоторую сумму денег. Как только над транзакцией осуществлен процесс майнинга в блоке с индексом i, злоумышленник может создать разветвление на цепочке блоков с индекса $i-1$, следовательно, изымая свою транзакцию новой серией блоков, которая в конечном итоге будет длиннее главной цепочки блоков. По мере того, как наиболее длинная ветвь становится принятой, транзакция злоумышленника не появляется в цепочке, так что, в конце концов, злоумышленник снимает некоторое количество денег, сохраняя свои монеты. Аналогичным методом, можно было бы легко переопределить блок, содержащий транзакцию от Алисы к Бобу. Возможность такой атаки возросла в контексте открытого регистра Bitcoin, поскольку было отмечено, что вычислительная мощность майнинга недостаточно рассеяна [18].

Предположение, что проблема совершения условных транзакций характерна исключительно для Pow-механизмов достижения консенсуса, поскольку существуют Blockchain-системы с другими механизмами достижения консенсуса, которые не подвержены подобной проблеме, является верным лишь частично. Это имеет место в случае некоторых Blockchain-систем, использующих механизм достижения консенсуса на основе подтверждения доли (proof-of-stake), таких как Tendermint, которые гарантируют согласие и достоверность консенсуса детерминировано. Однако проблема совершения условных транзакций применима и для Blockchain-систем, использующих PoS-механизм достижения консенсуса. Например, Casper представляет собой основанную на подтверждении доли альтернативу протоколу реорганизации GHOST, который используется в системе Ehtereum. Подтверждение доли не обязательно решает рассматриваемую проблему, поскольку даже Casper ставит в приоритет доступность по отношению к согласованности [19].

Другой проблемой является то, что реорганизация регистра может повлиять на исходный порядок транзакций. Это имеет значение при выполнении, в котором две транзакции нацелены на перевод $100 с одной учетной записи, первоначальный баланс которой составляет всего $100, поскольку может быть выполнена только транзакция, подтвержденная первой [20]. Однако проблема совершения условных транзакций является более общей по отношению к проблеме реорганизации регистра Blockchain. В частности, проблема совершения условных транзакций позволяет успешно выполнять конфликтующие транзакции и подтверждать их в двух разных состояниях цепочки блоков. Таким образом, решение проблемы, рассматриваемой в данной работе, позволит решить и проблему реорганизации регистра Blockchain.

Как известно, в асинхронной системе невозможно достичь консенсуса при наличии сбоев. Считается, что протокол гарантирует свойства или завершения, или соглашения детерминировано, но не оба эти свойства одновременно. В данной работе рассмотрено, что Blockchain консенсус завершается детерминировано на основе рекомендуемых 6-12 последующих добавленных блоков в Bitcoin [1] и Ethereum [16] соответственно, но иногда не достигает соглашения. Следует обратить внимание, что другие формализации также считают, что завершение достижения консенсуса в Bitcoin является детерминированным и что только его свойство безопасности является вероятностным [21]. Однако можно утверждать, что завершение не гарантируется детерминировано, а скорее вероятностно, и что можно увеличить вероятность достижения соглашения по консенсусу путем простого затягивания завершения. Характеристика консенсуса в Bitcoin-NG принимает это определение [22]. На практике, однако, Blockchain-приложения предполагают завершение достижения консенсуса, как описано в подразд. 2.2. Например, в Ethereum ожидание осуществления процесса майнинга над 12 последующими блоками объясняется тем, что, вероятно, такое ожидание будет достаточным для того, чтобы первый блок стал необратимым. Это может быть справедливо в крупномасштабной системе с открытым доступом, в которой вычислительная мощность майнинга достаточно рассеяна среди пулов майнинга, но его можно легко вернуть в контексте закрытой цепочки блоков.

Выводы

1. Проблема совершения условных транзакций предотвращает совершение пользователями наиболее популярных Blockchain-систем условных транзакций в случае разворачивания закрытого, частного регистра Blockchain. Данная проблема может привести к потере цифровых активов или атакам двойной траты. Возможным способом избегания проблемы совершения условных транзакций может быть создание смарт-контрактов вместо использования обычных транзакций, но это добавляет сложности и применимо только для систем с Тьюринг-полным языком программирования (например, Ethereum).

2. Если система достаточно велика и вычислительная мощность майнинга достаточно рассеяна среди достаточного количества пулов майнинга, то вероятность наличия пула майнинга, осуществляющего процесс майнинга быстрее других, может быть сделана сколь угодно малой. По этой причине рассматриваемая в работе проблема совершения условных транзакций имеет очень низкую вероятность появления в крупномасштабных свободно доступных Blockchain-системах.

3. В работе рассмотрены связи проблемы совершения условных транзакций с другими проблемами и показано, что она не ограничивается технологией Blockchain проекта Ethereum, но и потенциально может распространяться на частные, закрытые регистры Blockchain, использующие механизм достижения консенсуса proof-of-stake.

4. Blockchain-приложения требуют достижения консенсуса, чтобы быстро завершаться, в то время как базовые протоколы Blockchain гарантируют достижение консенсуса вероятностно. Это приводит к негативным результатам применительно к закрытым, частным распределенным регистрам.

5. В качестве направлений для последующих исследований можно выделить следующие: анализ проблемы совершения условных транзакций в Blockchain-системах, использующих PoS-механизм достижения консенсуса; изучение альтернативных Blockchain-систем, предоставляющих исключительно детерминированные гарантии для частной сети.

Список литературы: 1. *S. Nakamoto.* Bitcoin: a peer-to-peer electronic cash system, 2008. http://www.bitcoin.org. 2. *G. Wood.* Ethereum: A secure decentralised generalized transaction ledger final draft – under review, 2014. ttps://github.com/ethereum/wiki/wiki/White-Paper. 3. *M. J. Fischer, N. A. Lynch, and M. S. Paterson.* Impossibility of distributed consensus with one faulty process. J. ACM, 32(2):374–382, Apr. 1985. 4. *C. Dwork, N. Lynch, and L. Stockmeyer*. Consensus in the presence of partial synchrony. J. ACM, 35(2), Apr. 1988. 5. International Business Time Journal, 2015. http://www.ibtimes.co.uk/r3-connects-11-banksdistributed-ledger-using-ethereum-microsoft-azure-1539044. 6. *M. Castro and B. Liskov.* Practical byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst., 20(4):398–461, Nov. 2002. 7. *K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer.* On scaling decentralized blockchains. In 3rd Workshop on Bitcoin Research (BITCOIN), Barbados, February 2016. 8. *M. Vukol´*ıc. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In Proceedings of the IFIP WG 11.4Workshop on Open Research Problems in Network Security (iNetSec 2015), LNCS, 2016. 9. *C. Decker, J. Seidel, and R. Wattenhofer.* Bitcoin meets strong consistency. In Proceedings of the 17th International Conference on Distributed Computing and Networking (ICDCN), page 13, 2016. 10. *V. Gramoli, L. Bass, A. Fekete, and D. Sun.* Rollup: Nondisruptive rolling upgrade with fast consensus-based dynamic reconfigurations. IEEE Trans. on Parallel and Distributed Systems, 2015. 11. *L. Lamport.* The Part-Time parliament. ACM Transactions on Computer Systems, 16(2):133–169, May 1998. 12. *D. Ongaro and J. Ousterhout.* In search of an understandable consensus algorithm. In 2014 USENIX Annual Technical Conference (USENIX ATC 14), pages 305–319, Philadelphia, PA, 2014. USENIX Association. 13. *X. Xu, C. Pautasso, L. Zhu, V. Gramoli, S. Chen, A. Ponomarev, and A. B. Tran.* The blockchain as a software connector. In Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), April 2016. 14. *C. Dwork and M. Naor.* Pricing via processing or combating junk mail. In Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO ’92, pages 139–147, 1993. 15. *A. Black.* Hashcash – a denial of service countermeasure. Technical report, Cypherspace, 2002. 16. *G.Wood.* Ethereum: A secure decentralized generalized transaction ledger, 2015. Yellow paper. 17. QuadrigaCX Bitcoin Trading Platform. https://www.quadrigacx.com/. 18. *I. Eyal and E. G. Sirer.* Majority is not enough: Bitcoin mining is vulnerable. In Financial Cryptography and Data Security – 18th International Conference, FC 2014, hrist hurch, Barbados, March 3-7, 2014, Revised Selected Papers, pages 436–454, 2014. 19. *Ethereum* Stack Exchange. https://ethereum. stackexchange.com/questions/332/what-is-the-difference-between-casper-and-tendermint /536 20. *Wood G.* Chain Reorganization Depth Expectations, 2015. https://blog.ethereum.org/2015/08/08/ chain-reorganisation-depth-expectations/. 21. *J. A. Garay, A. Kiayias, and N. Leonardos.* The Bitcoin backbone protocol: Analysis and applications. In Advances in Cryptology – EUROCRYPT 2015 – 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II, pages 281–310, 2015. 22. *I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse.* Bitcoin-NG: A scalable blockchain protocol. In 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2016.

Харьковский национальный
университет радиоэлектроники

Поступила в редколлегию 29.03.2017