Биткоин: Пиринговая система цифровых активов .

Сатоси Накамото
satoshin@gmx.com
www.bitcoin.org
Перевод на русский Александр Шульгин
Alexander@shulgin.com
www.shulgin2035.expert

Абстракт. Исключительно одноранговая (пиринговая) версия цифровых активов позволяющая проводить электронные транзакции непосредственно от одной стороны к другой, без участия любых существующих финансовых учреждений. Цифровые подписи являются частью решения, но основная часть выгоды теряется, если доверенная третья сторона все еще необходима для предотвращения двойных расходов. Мы предлагаем решение проблемы двойных расходов с использованием одноранговой сети. Сеть фиксирует временные метки транзакций, формируя из них цепочку хешированных доказательств – проделанной-работы. Образованные таким образом записи, не могут быть изменены без повторного подтверждения работ по обработке данных. Самая длинная цепь не только служит доказательством последовательности событий, но и становится доказательством того, что она есть результат компьютерных вычислений самого большого по мощности пула процессоров. До тех пор, пока преобладающая часть вычислительных мощностей контролируется узлами, не объединенными для атаки сети, они будут генерировать самую длинную цепочку, опережая возможные атаки. Суть структуры самой сети достаточно проста. Сообщения транслируются с максимальными сжатые сроки на основе принципа наименьших затрат, усилиями и узлы могут покидать сеть и вновь присоединяться к ней по своему желанию, принимая самую длинную цепочку, как подтверждение произошедшего в цепочке в период такого отсутствия.

1. Введение

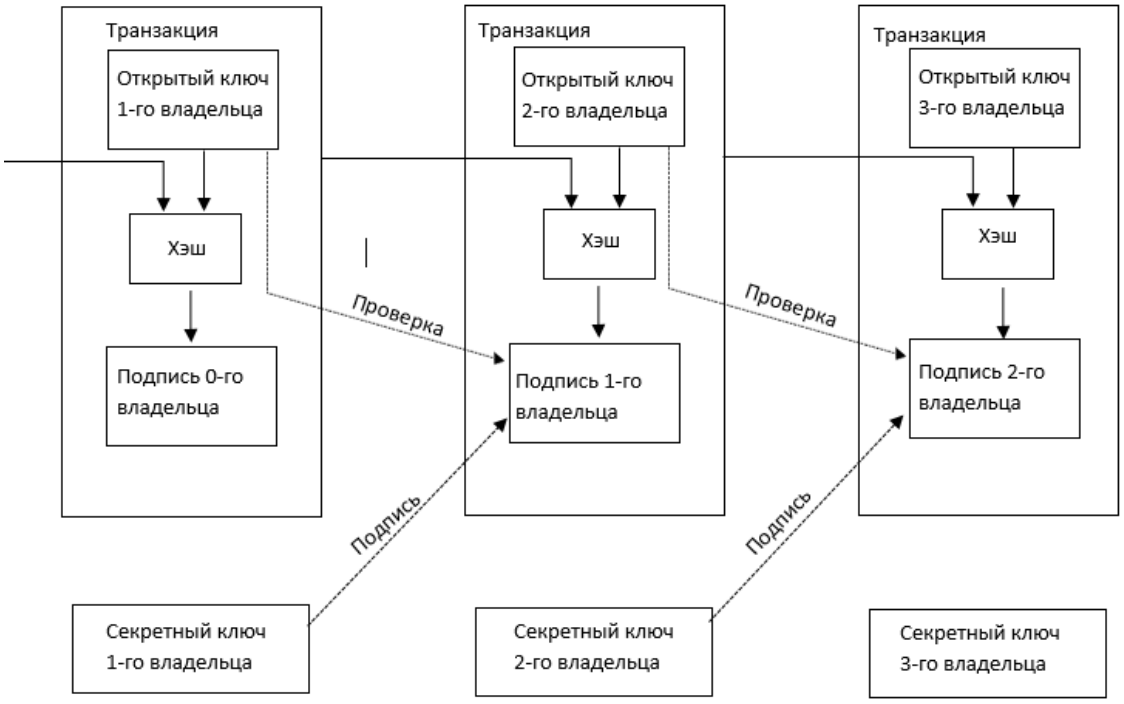
Электронная коммерция в цифровой среде почти во всех случаях обработки электронных платежей вынуждена полагаться на посредников, таких как финансовые учреждения, выступающие в качестве доверенных третьих лиц. Пока система работает достаточно хорошо для большинства сделок, но она все еще имеет недостатки, присущие модели, основанной на доверии. Полностью необратимые транзакции на самом деле невозможны, поскольку в посреднических спорах, невозможно обойтись без услуг финансовых учреждений. Стоимость такого посредничества увеличивает операционные издержки транзакций, ограничивает минимальный практический размер сделки и отсекает возможности для небольших транзакций. Также цену транзакций еще больше увеличивает отсутствие возможности производить необратимые платежи за неотменяемые услуги. При возможности обратимых транзакций нужда в доверии возрастает. Продавцы должны быть осторожны, остерегаясь своих клиентов, беспокоить их запрашивая большее чем необходимо количество информации. Определенный процент мошенничества принимается как неизбежность. Эти расходы и платежную неопределенность можно

избежать лично, используя физическую деньги, но нет механизма для осуществления платежей по электронным каналам связи без доверенной третьей стороны.

Всё, что нужно, это система электронных платежей, основанная на криптографических доказательствах, а не на доверии, которая позволяет любым двум желающим сторонам заключать прямые сделки друг с другом без необходимости посредничества третьей стороны. Сделки, которые для обратимости нецелесообразно компьютерно пересчитывать, защитят продавцов, от мошенничества, а для защиты покупателей можно легко внедрить обычные механизмы условного депонирования. В данном документе мы предлагаем решение проблемы двойного расходования с помощью распределенного однорангового сервера временных меток, который генерирует вычислительные подтверждения хронологического порядка транзакций. Система находится в зоне безопасности до тех пор, пока честные узлы суммарно контролируют больше вычислительных мощностей, чем любая совокупность атакующих узлов.

2. Сделки

Возьмём электронную монету как цепочку цифровых подписей. Очередной владелец передает монету последующему, подписывая хэш предыдущей сделки и публичный ключ следующего владельца и добавляя эту информацию к концу цепочки информации о монете. Получатель платежа может проверить подписи для проверки цепочки собственности



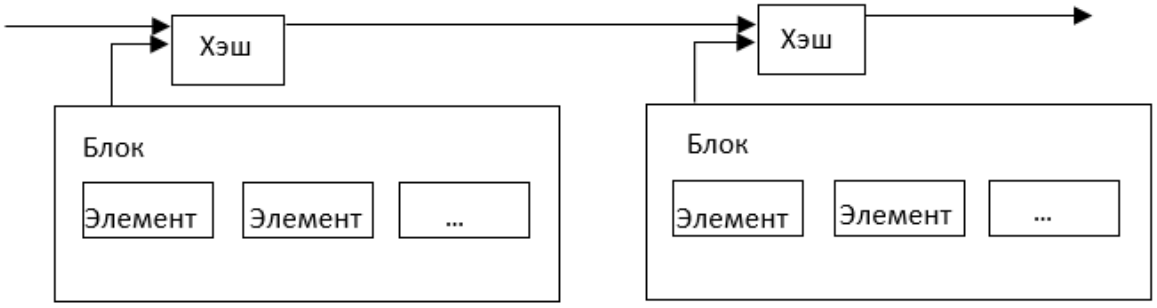
Проблема, конечно, в том, что получатель не может проверить, не расплатился ли предыдущий владелец этой монетой с кем-то ещё. Привычным решением является предоставление таких функций доверенному центральному органу или эмитенту, который проверяет каждую сделку на предмет двойного расходования. После каждой транзакции

монета должна быть возвращена эмитенту, который выпускает новую подобную монету, и только монеты, выпущенные непосредственно эмитентом, имеют полное доверие. Проблема такого решения заключается в том, что судьба всей денежной системы зависит непосредственно от компании - эмитента, и каждая сделка должна проходить непосредственно через такую компанию, точно так же как и в банке.

Нам нужен способ, чтобы получатель знал, что предыдущие владельцы не подписывали раньше никакие подобные сделки/ Для наших целей истинная сделка - это та, которая считается первой, поэтому нас не волнует о последующих попытках двойного расходования. Единственный способ подтвердить отсутствие сделки – это быть в курсе всех сделок. Поэтому в такой модели эмитент был осведомлен обо всех сделках и решал очередность транзакций. Чтобы избавится от посредника, сделки должны быть опубликованными [1], и нужна система, позволяющая участникам договориться о едином порядке истории транзакций. Получателю необходимо доказательство того, что в момент совершения каждой операции большинство узлов согласились считать её первой.

3. Сервер меток времени

Предлагаемое нами решение начинается с сервера временных меток. Его роль заключается в хэшировании из блока элементов, на которые будут ставиться временные метки, и открыто публиковать хэш, также как публиковали бы, например, в газете или Usenet постах [2-5]. Временная метка доказывает, что данные существовали в конкретное время, и потому и были включены в хэш. Каждая временная метка включает в себя предыдущую временную метку в своем хэше, образуя цепочку, с каждой дополнительной временной меткой, укрепляя все предыдущие в цепи.



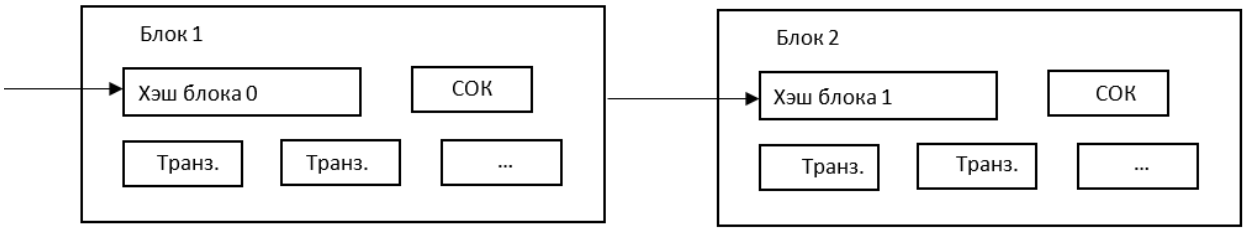
4. Доказательство проделанной работы.

Для реализации распределенного сервера временных меток на пиринговой основе нам потребуется использовать систему «доказательства-проделанной - работы», аналогичную системе Hashcash Адама Бэка [6], а не газеты и не Usenet-посты.

Доказательство-проделанной - работы являлся бы поиск значения, которое при хэшировании, например, с помощью SHA-256, будет давать хэш, который начинается с числа нулевых битов. Стандартный обьём выполняемой работы зависит от числа нулей, но для проверки найденного значения будет достаточно вычислить лишь один хэш

В нашем сервере меток времени поиск значения с нужным хэшем происходит путем перебора значения итерируемого поля - случайного одноразового кода (СОК) в

блоке данных. Как только блок, удовлетворяющий условию, найден, его содержимое нельзя изменить, не выполнив заново всей работы. И если он не является последним в цепочке, эта работа включает в себя и перевычисление всех блоков, следующих за ним.



Такое доказательство-проделанной работы также решает проблемы, определяя принятие решений большинством голосов. Если голосом считается один IP-адрес, то такую схему можно взломать, если контролировать распределенное большинство распределенных адресов. Принцип доказательства-работы является, по сути, «один процессор - один голос». Самая длинная из хэш-цепочек выражает мнение большинства, которое вложило в нее наибольшее количество ресурсов. Если более половины вычислительной мощи контролируется честным узлам, то цепочка честных транзакций будет расти быстрее и опередит любую конкурирующую цепь. Чтобы внести изменения в любой из прошлых блоков, атакующему сеть придется выполнить и предъявить заново доказательство-проделанной работы над этим блоком и всеми последующими, а затем догнать и перегнать честных участников по новым блокам. Ниже мы покажем, что вероятность такого успеха у злоумышленника, обладающего меньшими ресурсами, уменьшается в геометрической прогрессии в зависимости от числа блоков.

Чтобы компенсировать увеличение вычислительной мощи процессоров и колебания работающих узлов в сети, сложность хэширования должна изменяться, чтобы обеспечивать равномерную скорость генерации блоков в час. Если они появляются слишком быстро — сложность возрастает, и наоборот.

5. Сеть

Правила работы сети:

1) Новые транзакции рассылаются всем узлам.
2) Каждый узел собирает пришедшие транзакции в блок.
3) Каждый узел подбирает подходящую сложность для доказательство-проделанной-работы для своего блока.
4) Как только такой хэш доказательство-проделанной работы найден, этот блок отправляется в сеть на все узлы.
5) Узлы принимают блок, только если все транзакции в нем корректны и истрачены полностью.
6) Свое согласие с новыми данными узлы выражают, работая над созданием следующего блока в цепи, и используя хэш предыдущего блока в качестве основы.

Самая длинная цепочка всегда будет считаться правильной, и узлы будут работать над ее удлинением. Если два узла одновременно опубликуют разные версии очередного блока, то кто-то из остальных узлов получит раньше одну версию, а кто-то — другую. В

таком случае узлы будут работать над тем узлом, который был получен первым, но сохраняя другую ветку цепи, на случай если она станет длиннее. Окончательное решение придет, как только будет получено доказательство-проделанной-работы для нового блока, который продолжит любую из ветвей, и те узлы, что работали над спорной веткой цепи, переключатся на верную.

Новые транзакции не обязательно должны дойти до всех узлов. Как только они достигнут достаточное множество узлов, то значит вскоре они попадут в один из блоков. Принципы коммуникаций блоков толерантны в отношении потерянных сообщений. Как только узел, пропустивший один из блоков, получит уже следующий за ним, и понимая, что он пропустил предыдущий, он запросит недостающую информацию, чтобы заполнить пробел в цепи.

6. Стимулы

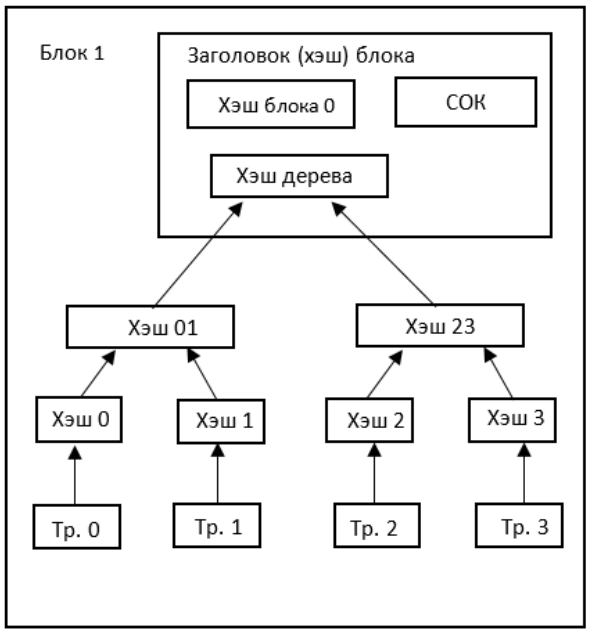
По условию, первая транзакция в блоке - это специальная сделка, с которой начинается новая монета, принадлежащая создателю блока. Это добавляет мотивацию для узлов поддерживать сеть и обеспечивает способ первоначального введения монет в обращении, без какого либо централизованного контролирующего органа-эмитента. Стабильная добыча и введение постоянного количества новых монет имеет аналогии с золотодобытчиками, которые также инвестировали свои средства и ресурсы для добычи золота и включении его в коммерческий оборот. В нашем случае такими средствами и ресурсами являются процессорная мощность и время и, конечно же, электричество.

Иным стимулом может быть комиссия за транзакции. Если входная сумма платежа больше выходной, то разница является комиссией за перевод и прибавляется к базовому значению награды за найденный блок в первой транзакции. Как только суммарный объем денежной массы достигнет заранее установленного максимума, единственным источником поощрения работы над блоками останутся комиссии, при этом не подверженными инфляции.

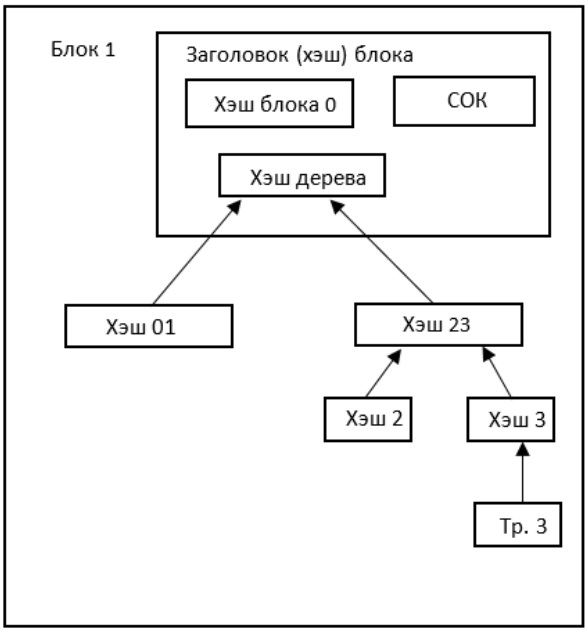
Мотивации могут также стимулировать узлы оставаться честными. Если атакующая сторона, ведомая жадностью способна собрать больше вычислительных мощностей, чем все честные участники, то придется выбрать между воровством, аннулируя свои транзакции у продавцов и крадя средства, или же направить такие свои ресурсы на добычу новых блоков и монет. Более выгодным для всех является вариант «игры по правилам», который обеспечивает получение больше новых монет, чем все остальные, нежели подрыв системы и как следствие, и своего собственного капитала.

7. Оптимизация дискового пространства

Как только последняя транзакция в монете будет встроена в достаточное количество предыдущих блоков, все предшествующие ей транзакции в цепочке могут быть удалены в целях очистки дискового пространства. Чтобы хэш блока остался неизменным, все транзакции в блоке хранятся в виде хэш-дерева Меркла [7][2][5] и лишь его корень включается в хэш блока. Размер старых блоков может быть уменьшен за счет удаления ненужных ветвей этого дерева, хранить промежуточные хэш необязательно.



Все транзакции в дереве Меркла



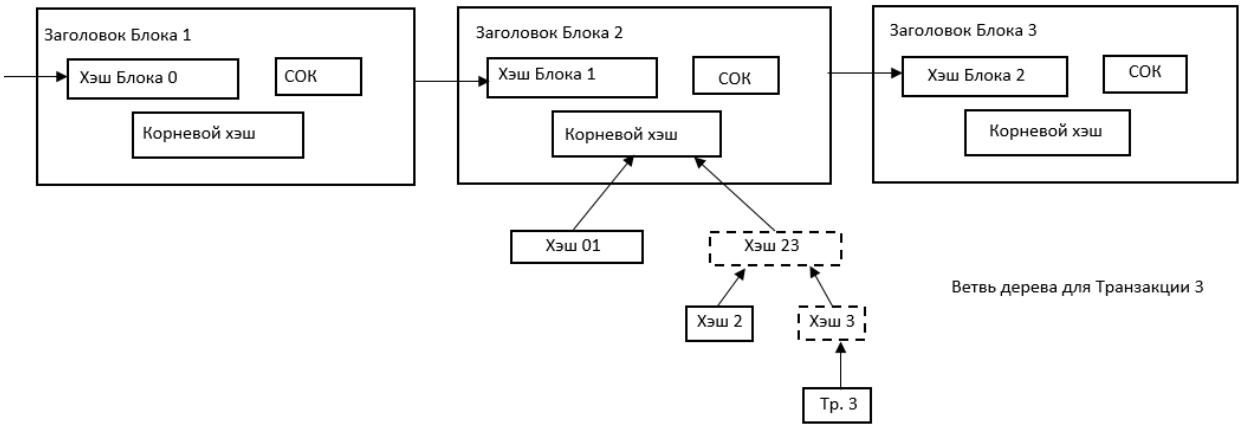
После обрезки Тр. 0-2 из Блока 1

Заголовок пустого блока будет составлять около 80 байт. Из расчета скорости генерации блока раз в десять минут получаем 80*6*24*365=4.2 Мб в год. Для среднестатистического на 2008 год компьютера с 2 Гб оперативной памяти с учетом закона Мура, предсказывающего рост на 1.2 Гб в год, хранение данных не будет проблемой, даже если все заголовки блоков будут находиться в памяти.

8. Упрощенная проверка платежа

Верификация транзакций возможна и без запуска полнофункционального узла. Пользователю необходимо лишь хранить заголовки блоков самой длинной цепочки доказательства - проделанной-работы, которую он получил от других узлов, и запрашивать ветку Меркла для необходимой транзакции в блок с временными метками. Он не может проверить корректность транзакции самостоятельно, но получив ссылку на блок, в котором она находится, он может убедиться в том, что этот блок и все последующие приняты и подтверждены сетью.

Самое длинное доказательство работоспособности цепочки

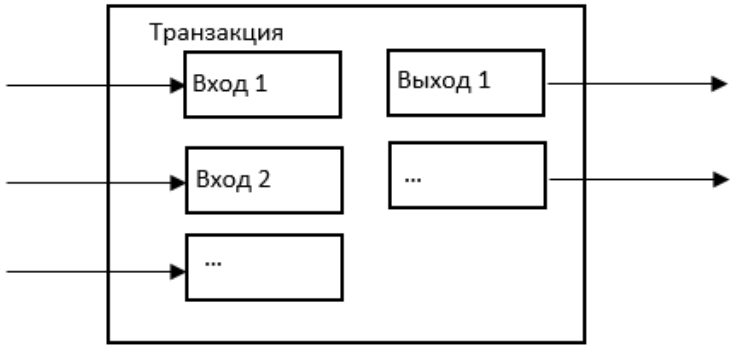


Ветвь дерева для Транзакции 3

На такой метод проверки можно полагаться, пока сеть хотя бы наполовину находится под контролем честных участников, то есть пока сетевые атаки не завладели преобладающими ресурсами сети. Обычные узлы могут проверять транзакции самостоятельно, но если нападающий генерирует самую длинную цепь блоков, то своими сфабрикованными транзакциями он может продолжать подчинять сеть. Одной из стратегий противодействия этому может быть рассылка сигналов тревоги от обычных пиров, которые получают «ложный» блок. Такой сигнал будет заставлять программное обеспечение пользователя загружать блок полностью, чтобы самостоятельно подтверждать несоответствие данных. Лица, часто принимающие платежи, возможно, будут подключаться к сети в обычном режиме для большей независимости, безопасности и быстроты проверки.

9. Объединение и разъединение сумм.

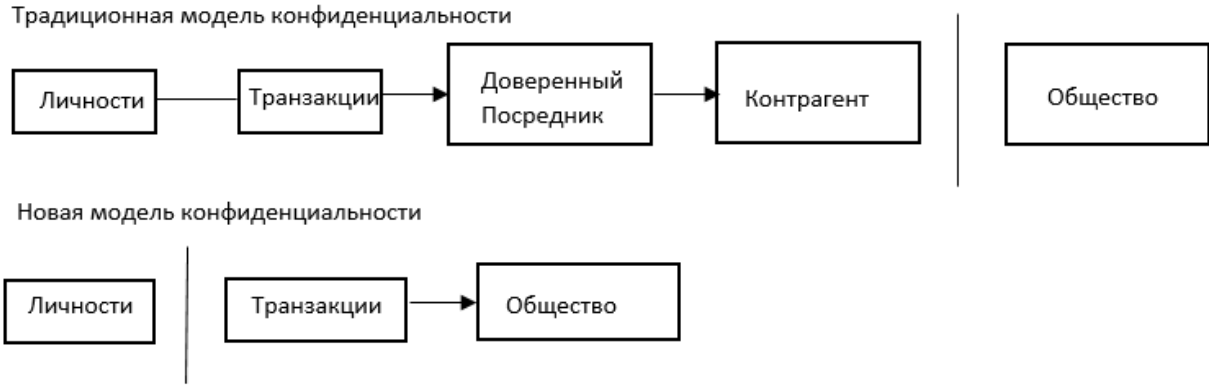
Монетами можно обращаться по отдельности, но было бы неразумно делать отдельные операции для перевода каждого цента. Позволяя разделить и объединить объемы перевода, сами транзакции содержат несколько входов и выходов. Обычно будет либо один вход от более крупной предыдущей операции или множественных входов, объединяющих меньшие суммы, и не более двух выходов: один для оплаты, и другой для возврата разницы , если такая возникнет, обратно отправителю.



Следует отметить, что разветвление связей, когда транзакция зависит от нескольких, а те в свою очередь зависят от еще большего числа, не является проблемой, поскольку нет необходимости получать полную и независимую копию истории транзакции.

10. Конфиденциальность

Традиционная банковская модель придерживается такого уровня конфиденциальности, где доступ к информации предоставляется лишь вовлеченным сторонам и доверенному посреднику. Необходимость публично объявлять обо всех сделках исключает этот метод, но конфиденциальность по-прежнему может быть сохранена путем сохранения анонимности открытых ключей. Публичной будет информация, что кто-то посылает кому-то другому некоторую сумму, но без информации, связывающей конкретную сделку с конкретной личностью. Это аналогично уровню информации, публикуемой фондовыми биржами, где время и объем индивидуальных сделок открыто публикуются, но без указания того, кто именно являлся сторонами таких сделок



Дополнительной межсетевой защитой будет являться генерация новой пары «открытый/закрытый ключ» для каждой транзакции: это предотвратит связывание различных платежей с их общим отправителем или адресатом. Некоторого публичного связывания все же не избежать: транзакции с несколькими входами доказывают, что эти суммы принадлежат одному лицу. Риск заключается в том, что раскрытие личности владельца ключа может выявить и другие принадлежащие ему сделки.

11. Расчеты

Мы рассматриваем сценарий, когда при сетевой атаке идет попытка сгенерировать альтернативную цепь быстрее, чем честные участники. Даже если это и удастся, это не даст шанс делать деньги из воздуха или присваивать чужие монеты и ценности или вообще вносить какие либо изменения. Узлы не примут в качестве оплаты недействительную сделку, ровно как никогда не примут блок, содержащий такую транзакцию.

При сетевой атаке могут только попытаться изменить одну из своих транзакций, чтобы взять обратно деньги, которыми недавно свершили оплату.

Гонку между честной цепочкой и атакующей цепочкой можно представить как биномиальное случайное блуждание. Успешное событие, когда честная цепь удлиняется на один блок, приводит к увеличению отрыва на +1, а неуспешное, когда очередной блок создает атакующая цепь, что ведет к сокращению разрыва -1. Вероятность успеха для атакующего наверстать разрыв в несколько блоков такая же, как и для в задаче о «разорении игрока». Представим, что игрок имеет неограниченный кредит, начинает с некоторым дефицитом и у него есть бесконечно много попыток, чтобы отыграться.

Вероятность того, что он преуспеет, как и вероятность злоумышленника догнать честных участников, вычисляется следующим образом [8]:

p = вероятность появления блока в честной цепочке

q = вероятность того, что блок создаст атакующий

q_z= вероятность того, что атакующий наверстает разницу в z блоков

$$q_z = \begin{cases} 1 & if\ p \leq q \\ (q/p)^z & if\ p > q \end{cases}$$

В случае $p > q$ вероятность уменьшается экспоненциально с ростом числа блоков, на которое отстает злоумышленник. Поскольку все коэффициенты вероятности против него, то без удачного рывка вначале, его шансы на успех становятся ничтожны. Рассмотрим теперь, как долго получателю платежа стоит ждать, прежде чем он будет полностью уверен, что бывший владелец не сможет отменить транзакцию. Мы предполагаем, что злоумышленник-отправитель позволяет адресату некоторое время верить, что платеж был проведен, после чего возвращает деньги себе. Получатель узнает об этом, но мошенник надеется, что будет уже слишком поздно. Получатель создает новую пару ключей и сообщает свой публичный ключ отправителю прямо перед подписанием транзакции. Это не позволит отправителю заранее начать работать над цепочкой и провести транзакцию в тот момент, когда он будет достаточно удачлив, чтобы совершить рывок вперед. После отправки платежа мошенник начинает втайне работать над параллельной версией цепочки, содержащей альтернативную транзакцию. Получатель ждет, пока транзакция не будет добавлена в блок и пока тот не будет продолжен еще z блоками. Ему неизвестен прогресс злоумышленника, но если средняя скорость генерации честных блоков — известная величина, то число блоков нападающего подчиняется распределению Пуассона с математическим ожиданием:

$$\lambda = z\frac{q}{p}$$

Чтобы получить вероятность того, что атакующий может догнать в моменте, мы умножаем увеличенное значение случайной величины на вероятность того, что он сможет сократить оставшуюся разницу:

$$\sum_{k=0}^{\bowtie} \frac{\lambda^{\kappa} e^{-\lambda}}{\kappa!} \cdot \begin{cases} (q/p)^{(z-k)} & if\ k \leq z \\ 1 & if\ k > z \end{cases}$$

Поменяв местами слагаемые и избавившись от бесконечного ряда, получаем:

$$1 - \sum_{k=0}^{z} \frac{\lambda^{\kappa} e^{-\lambda}}{\kappa!} \cdot \left(1 - (q/p)^{(z-k)}\right)$$

Конвертировав в код Си получили следующее:

```
#include <math.h> double AttackerSuccessProbability(double q, int z)
{
        double p = 1.0 - q;
        double lambda = z * (q / p);
        double sum = 1.0;
        int i, k;
        for (k = 0; k <= z; k++)
        {       double poisson = exp(-lambda);
                for (i = 1; i <= k; i++)
                poisson *= lambda / i;
                sum -= poisson * (1 - pow(q / p, z - k));
        }
        return sum; }
```

Просматривая некоторые результаты, мы видим, что вероятность падает экспоненциально с z.

```
q=0.1
z=0   P=1.0000000
z=1   P=0.2045873
z=2   P=0.0509779
z=3   P=0.0131722
z=4   P=0.0034552
z=5   P=0.0009137
z=6   P=0.0002428
z=7   P=0.0000647
z=8   P=0.0000173
z=9   P=0.0000046
z=10  P=0.0000012

q=0.3
z=0   P=1.0000000
z=5   P=0.1773523
z=10  P=0.0416605
z=15  P=0.0101008
z=20  P=0.0024804
z=25  P=0.0006132
z=30  P=0.0001522
z=35  P=0.0000379
z=40  P=0.0000095
z=45  P=0.0000024
z=50  P=0.0000006
```

Решение для P менее 0,1%...

P < 0.001
q=0.10 z=5
q=0.15 z=8
q=0.20 z=11
q=0.25 z=15
q=0.30 z=24
q=0.40 z=89
q=0.45 z=340

12. Заключение.

Мы предложили систему электронных транзакций, не основанную на доверии. Построение структуры началось с традиционного представления монет на основе цифровых подписей, обеспечивающего контроль владения, но допускающего двойную трату. Эту проблему мы решили посредством пиринговой сети и схемы «доказательства – проделанной- работы» для записи публичной истории транзакций. Попытки атак на сеть, не обладая большей частью ресурсов сети, изменить старые записи, вычислительно становится практически неосуществимым, если честные узлы контролируют большую часть мощности сети. Сеть надежна в своей неструктурированной простоте. Все узлы работают самостоятельно с небольшой координацией между собой. Их не нужно идентифицировать, так как сообщения не направляются в какое-либо конкретное место, и доставляются только следуя принципам «наименьших затрат».

Узлы могут оставлять сеть и вновь подключаться, принимая самую длинную цепочку блоков как подтверждение пропущенной истории транзакций. Они подтверждают свое согласие принять корректный блок в цепочку, используя свои вычислительные мощности процессоров для удлинения этой цепи, или несогласие, отвергая некорректные блоки, отказываясь от работы с такими блоками. С помощью такого механизма консенсуса можно обеспечить соблюдение любых необходимых правил и стимулов.

Ссылки на источники

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In *20th Symposium on Information Theory in the Benelux*, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In *Journal of Cryptology*, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In *Sequences II: Methods in Communication, Security and Computer Science*, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.