Система онлайн-голосования Polys

Обзор
Версия 2.0

kaspersky АКТИВИРУЙ БУДУЩЕЕ

Узнать больше
на kaspersky.ru

Краткое содержание

В современном обществе все активнее обсуждается идея онлайн-голосования. Онлайн-голосование обладает высоким потенциалом как способ уменьшить организационные расходы и повысить активность избирателей. При такой системе не нужно печатать бумажные бюллетени и открывать избирательные участки. Участники голосования могут проголосовать откуда угодно, где бы они ни находились: все, что им необходимо, это подключение к интернету. Несмотря на названные достоинства такого голосования, к решениям для онлайн-голосования относятся очень осторожно, так как с ними сопряжены новые угрозы, и даже единственная уязвимость может способствовать осуществлению крупномасштабных манипуляций.

Принципы онлайн-голосования изучаются на протяжении уже нескольких десятилетий. За это время разработано множество методик, которые в определенной степени справляются с задачей создания систем онлайн-голосования. Тем не менее, обеспечение всех требуемых характеристик в системе голосования является далеко не тривиальной задачей, поэтому разработчики таких систем обычно вынуждены идти на некоторый компромисс. При этом важной задачей является создание такой системы, которая была бы прозрачной для всех и которая позволяла бы осуществлять аудит и верификацию каждого из процессов. В нашем решении для онлайн-голосования используются передовые методики, позволяющие получить желаемые характеристики. В его основе лежит технология блокчейн, которая обеспечивает максимальную верификацию для всех пользователей системы.

Проблемы создания систем онлайн-голосования и их решение

- Контроль за наличием активного избирательного права
- Невозможность многократного участия
- Конфиденциальность
- Недоступность промежуточных результатов
- Корректность результатов
- Полнота

Идет ли речь о традиционной процедуре голосования с использованием бумажных бюллетеней, голосовании с использованием специальных технических средств (КОИБ[1] / КЭГ[2]) или же об онлайн-голосовании, необходимо, чтобы система голосования удовлетворяла следующим требованиям:

- Контроль за наличием активного избирательного права (eligibility): к голосованию допускаются только те, кто наделен соответствующим правом.
- Невозможность многократного участия (unreusability): каждый может проголосовать только один раз.
- Конфиденциальность (privacy): информация о выборе голосующего не может быть доступна никому, кроме него самого.
- Недоступность промежуточных результатов (fairness): промежуточные итоги голосования недоступны никому.
- Корректность результатов (soundness): необходимо выявить недействительные бюллетени и не учитывать их при итоговом подсчете.
- Полнота (completeness): все бюллетени, признанные действительными, должны быть правильно посчитаны.

Далее мы приводим краткий обзор решений для систем онлайн-голосования, которые удовлетворяют перечисленным выше требованиям.

Решение вопроса о наличии права голоса вполне очевидно: чтобы участвовать в онлайн-голосовании, голосующие должны пройти процедуру идентификации личности с помощью доверенной системы идентификации. При этом идентификационные данные легитимных избирателей должны быть внесены в список участников голосования. Однако тут возникают серьезные проблемы. Во-первых, все изменения в списке участников голосования подлежат проверке, чтобы в нем не появились нелегитимные избиратели. Во-вторых, система идентификации должна пользоваться доверием и быть защищенной, чтобы доступ к учетной записи голосующего пользователя не получил злоумышленник. Создать систему идентификации, которая обладала бы подобными свойствами, — это уже само по себе непростая задача. Тем не менее такие системы необходимы для целого ряда иных видов деятельности, особенно тех, что относятся к цифровым услугам государственных органов. И поэтому мы считаем, что оптимально будет применить существующую систему идентификации и оставить вопрос создания таковой за пределами нашего исследования.

1 КОИБ – комплекс обработки избирательных бюллетеней. Данное устройство сканирует, распознает и производит подсчет бумажных бюллетеней.

2 КЭГ – комплекс электронного голосования, представляет собой устройство для голосования с сенсорным экраном.

Реализовать невозможность многократного участия, на первый взгляд, очень просто: необходимо всего лишь отмечать в списке тех участников, которые уже отдали свой голос, и не допускать их до голосования повторно. Однако при этом не стоит забывать о конфиденциальности, так что обеспечить невозможность многократного участия и одновременно анонимность голосования может оказаться трудной задачей. Более того, может потребоваться разрешить участнику голосования поменять свое решение и переголосовать. И тогда задача усложняется еще больше. Краткий обзор методов, при помощи которых реализуют невозможность многократного участия, будет приведен далее, параллельно с обзором на тему обеспечения конфиденциальности.

Слепая подпись — это способ подписи данных, при котором подписывающая сторона не знает, что именно подписывает.

Конфиденциальность в контексте онлайн-голосования означает, что никто, кроме самого голосующего, не знает, как он проголосовал. Это достигается в основном с помощью одной (или нескольких) из следующих технологий: слепая подпись, гомоморфное шифрование и смешанные сети. Слепая подпись — это способ подписи данных, при котором подписывающая сторона не знает, что именно подписывает. Это достигается за счет использования функций ослепляющего шифрования и подписи, которые коммутативны: *Blind(Sign(message)) = Sign(Blind(message))*. Запрашивающая сторона ослепляет сообщение (применяет к нему функцию ослепляющего шифрования) и отправляет его другой стороне на подпись. Получив подпись для ослепленного сообщения и зная фактор ослепления, она получает подпись уже для неослепленного сообщения. Слепые подписи математически исключают, что кто-либо, кроме инициатора процесса подписания, получит возможность связать ослепленное сообщение и соответствующую ему подпись с неослепленным сообщением и подписью для этого неослепленного сообщения.

В протоколе голосования, предложенном в 1992 году Фудзиокой, Окамото и Охтой, схема слепой подписи применяется следующим образом: легитимный избиратель выполняет ослепляющее шифрование своего бюллетеня и отсылает его валидатору. Валидатор удостоверяется в том, что избиратель имеет право участвовать в голосовании, подписывает ослепленный бюллетень и возвращает его обратно избирателю. Далее избиратель снимает ослепляющее шифрование с бюллетеня, подписанного валидатором, и отправляет его в сервис подсчета голосов, который удостоверяет подпись валидатора, прежде чем принять бюллетень.

Эта схема лежит в основе множества протоколов онлайн-голосования. По сравнению с исходной схемой новые системы голосования удобнее в использовании (к примеру, в исходной схеме избирателю приходится ждать окончания выборов и отправлять ключ расшифрования бюллетеня), в некоторых из этих систем добавляется возможность переголосования и противодействие принуждению. Главная угроза в данном случае связана с подписывающей стороной: необходим доступный для проверки журнал всех выданных подписей. Эта информация логически связана с процедурой выдачи избирателю бюллетеня, поэтому необходимо удостовериться, что подписывающая сторона выдает подписи только легитимным избирателям. Также должно быть подтверждено, что учетные записи избирателей, которые имеют право участвовать в голосовании, но не воспользовались им, не могут быть использованы злоумышленником. Чтобы по-настоящему разорвать связь между избирателем и бюллетенем, для пересылки бюллетеня вместе с подписью необходимо использовать анонимный канал.

Гомоморфное шифрование — это тип шифрования, который позволяет производить математические операции над зашифрованными данными без их расшифрования.

Гомоморфное шифрование — это тип шифрования, который позволяет производить математические операции над зашифрованными данными без их расшифрования, например: *Enc(a) + Enc(b) = Enc(a + b)* — сложение; *Enc(a) * Enc(b) = Enc(a * b)* — умножение. С точки зрения онлайн-голосования схема шифрования, гомоморфная по сложению, позволяет получить сумму голосов всех избирателей в зашифрованном виде и расшифровать итоговый результат.

Здесь уместно заметить, что схема, гомоморфная по умножению, в принципе может использоваться как гомоморфная по сложению. Например, если имеются варианты выбора *x* и *y* и мы применяем схему шифрования, гомоморфную по умножению, мы можем взять значение *g* и зашифровать это значение, возведенное в степень: $Enc(g^x) * Enc(g^y) = Enc(g^{(x+y)})$.

Гомоморфное шифрование можно использовать, чтобы обеспечить наличие различных свойств, необходимых в системе онлайн-голосования. Так, в отношении конфиденциальности его используют таким образом, чтобы расшифрованию подлежала исключительно сумма всех голосов избирателей, но ни в коем случае не голос каждого в отдельности. На практике применение гомоморфного шифрования для обеспечения конфиденциальности предполагает, что расшифрование выполняется несколькими доверенными сторонами, чтобы никто полностью не владел ключом расшифрования. В противном случае конфиденциальность будет нарушена.

Обычно этот подход реализуется в виде пороговой схемы шифрования. Предположим, есть *n* субъектов. Чтобы расшифровать результаты, необходимо *t* субъектов, *t* <= *n*. Согласно схеме, каждый из субъектов применяет свою часть ключа к сумме зашифрованных результатов голосования, и когда эту операцию выполнят *t* субъектов, можно будет получить общую сумму голосов в расшифрованном виде. В отличие от схемы слепой подписи, анонимный канал между избирателем и системой не требуется, но конфиденциальность зависит от степени доверия к этим субъектам: в случае намеренного сговора существует угроза деанонимизации избирателей.

Смысл смешанных сетей в том, что голоса избирателей проходят через несколько прокси-серверов с выполнением расшифрования или перешифрования.

Смешанные сети также основаны на распределении доверия, но в ином виде. Смысл смешанных сетей в том, что поданные избирателями голоса проходят через несколько прокси-серверов, которые перемешивают их и выполняют некоторое действие — расшифрование или перешифрование, в зависимости от типа смешанной сети. В смешанной сети с расшифрованием у каждого сервера есть собственный ключ, и избиратель последовательно, как бы слой за слоем, зашифровывает свой голос на открытых ключах серверов. В результате получается своеобразная «луковица», с которой каждый сервер будет снимать определенный слой шифрования. В смешанной сети, использующей перешифрование, каждый узел сети производит повторное зашифрование голосов избирателей.

Существует множество вариантов смешанных сетей; полный обзор их свойств выходит за рамки данного документа. В отношении конфиденциальности здесь важно то, что теоретически, если хотя бы один из узлов смешанной сети выполняет смешивание честно, конфиденциальность соблюдается. В этом состоит некоторое отличие от конфиденциальности, получаемой при гомоморфном шифровании, когда мы делаем допущение, что некоторые из субъектов могут действовать из злого умысла. Принцип действия смешанных сетей может применяться для создания анонимных каналов, необходимых в других технологиях.

Недоступность промежуточных результатов достигается очевидным способом: прежде чем отправить свой голос, голосующий зашифровывает его, и голос расшифровывается только по окончании процесса голосования. В данном случае важно помнить, что если у кого-либо имеется ключ расшифрования с доступом к зашифрованным голосам, он может получить информацию о промежуточных результатах. Чтобы исключить эту проблему, ключ распределяется между несколькими хранителями. Система, где для расшифрования необходимы все хранители ключа, не является надежной: если один из них не участвует в процессе расшифрования, оно становится невыполнимым. Поэтому используются пороговые схемы, когда для расшифрования требуется участие лишь определенного числа хранителей ключа. Существуют два основных метода распределения ключа. Первый метод предполагает разделение секрета, когда доверенное лицо создает ключ, разделяет его на части и распределяет их между хранителями (например, схема Шамира). Второй метод предполагает распределенную генерацию ключей, когда нет необходимости в доверенном лице, а в вычислении ключа участвуют все стороны (например, протокол распределенной генерации ключей, описанный Педерсеном).

Доказательство с нулевым разглашением — это криптографический метод доказательства утверждения о значении без раскрытия самого значения.

На первый взгляд, требования полноты и корректности результатов вполне очевидны, однако их реализация может быть сопряжена с проблемами — в зависимости от протокола. Если бюллетени расшифровываются последовательно, один за другим, отличить действительные от недействительных несложно. Но в случае использования гомоморфного шифрования ситуация усложняется. Поскольку отдельно взятый бюллетень не подлежит расшифрованию, результат расшифрования не покажет тех случаев, когда в бюллетене было выбрано более одного варианта или если бюллетень был сформирован так, что воспринимается сразу как 10 (или миллион) голосов. По этой причине необходимо подтвердить, что зашифрованные данные соответствуют характеристикам действительного бюллетеня, не раскрывая никакой информации, которая может помочь выяснить, какой именно был подан голос. Эта задача решается посредством доказательств с нулевым разглашением. Согласно определению, это криптографический метод доказательства утверждения о значении этого утверждения без раскрытия значения как такового. В частности, в таких случаях, чтобы доказать, что конкретное значение относится к определенному набору данных, используются доказательства принадлежности диапазону (zero-knowledge range proofs).

Описанные выше свойства — это самый минимум, необходимый для любой системы голосования. Но все перечисленные технологии бесполезны, если сама система не пользуется доверием. Чтобы заработать это самое доверие, система голосования должна быть полностью доступной для проверки. То есть каждый, кто в ней задействован, должен иметь возможность убедиться, что система соответствует заявленным характеристикам. Задачу по обеспечению

проверяемости можно разделить на две: персональную, когда избиратель может удостовериться, что его бюллетень записан и учтен правильно, и всеобщую, когда каждый может проверить, что система в целом работает как полагается. Это предполагает публикацию входных и выходных данных различных этапов протокола голосования наряду с доказательством того, что он работает правильно. Например, принцип работы смешанных сетей полагается на доказательство правильного смешивания (разновидность доказательства с нулевым разглашением), а доказательство правильного расшифрования также используется в смешанных сетях и для порогового расшифрования. Очевидно, что чем больше процесс открыт для общественного контроля, тем выше доступность системы для проверки. Однако в онлайн-голосовании широко используется криптография, и чем выше сложность криптографических алгоритмов, тем менее понятной оказывается система для большинства пользователей. Немалое время может потребоваться на то, чтобы изучить протокол, и еще больше — на то, чтобы выявить в нем какую-либо уязвимость или бэкдор. И даже если вся система в целом тщательно изучена, все равно необходимо каким-либо способом гарантировать аутентичность выполняемого кода в процессе голосования.

Одно из желаемых свойств системы онлайн-голосования — противодействие принуждению.

Наконец, не менее важное значение имеют такие проблемы, как принуждение и покупка голосов. Онлайн-голосование выводит эти вопросы на новый уровень: ввиду того что бюллетени подаются дистанционно и из неконтролируемой среды, действия по оказанию давления или влияния на результат путем подкупа могут принимать масштабный характер. Вот почему одним из целевых свойств системы онлайн-голосования является противодействие принуждению. Противодействием это называется потому, что необходимо ограничить возможность действовать у любого, кто хочет повлиять на голосующего, руководить им и контролировать его действия. Смысл в том, чтобы принять максимально возможные меры, призванные уменьшить риск масштабного вмешательства. Оба типа нарушителей: и оказывающие давление, и покупающие голоса — требуют от избирателя подтверждения того, как именно он проголосовал. По этой причине во многих схемах голосования, где реализуется противодействие принуждению, используется принцип отсутствия подтверждения (receipt-freeness): избиратель не может доказать, что он проголосовал так или иначе. Подход к реализации этого принципа, как правило, основан на применении доверенного средства — системы или устройства, скрывающих уникальные параметры, которые используются для создания бюллетеня, отправляемого избирателем. Таким образом, избиратель не может подтвердить, что конкретный бюллетень принадлежит именно ему. У такого подхода есть и обратная сторона — если избиратель утверждает, что его бюллетень зарегистрирован или посчитан неправильно, он попросту не может подтвердить это ввиду отсутствия доказательств.

В нашем кратком обзоре технологий мы рассматривали каждую из них в отдельности. Тогда как в реальности в большинстве систем голосования при интеграции нескольких технологических решений для выполнения требований, предъявляемых к таким системам, приходится искать некоторый компромисс.

Например, как мы отметили, когда описывали слепую подпись, существует риск, что доступ к голосованию получат пользователи, не имеющие права участвовать в нем. Принцип отсутствия подтверждения вступает в противоречие с принципом проверяемости системы; более сложный протокол может существенно понизить удобство использования и т. д. Кроме того, мы рассмотрели лишь основные принципы построения решения, однако в реальном мире при создании системы требуется учесть значительно больше аспектов. Это и безопасность, и надежность сетевых протоколов, и процесс развертывания системы, и доступ к системным компонентам. В настоящее время не существует такого протокола, который обладал бы абсолютно всеми желаемыми свойствами. И следовательно, пока отсутствует система онлайн-голосования, которая была бы надежной на все 100%.

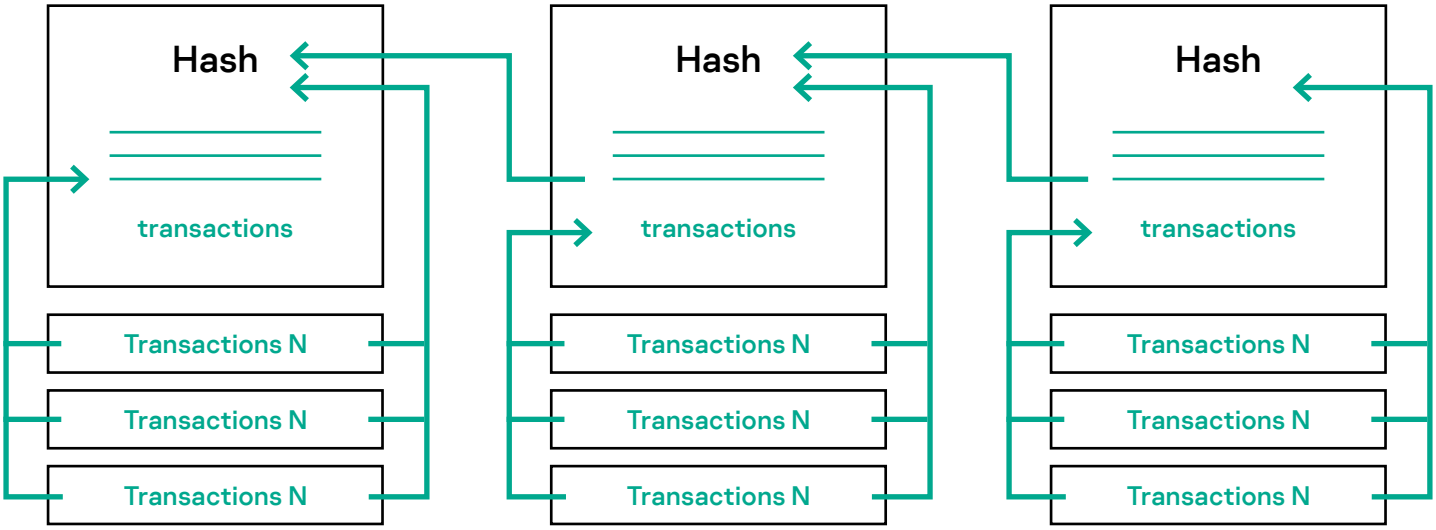
То, что мы сегодня называем блокчейном, — это совокупность технологий, объединенных вместе: сама структура данных блокчейна, алгоритм распределенного консенсуса, криптография с открытым ключом и смарт-контракты.

Технология блокчейн

При упоминании технологии блокчейн первым делом на ум приходят криптовалюта и смарт-контракты – благодаря таким широко известным проектам, как Bitcoin и Ethereum. Bitcoin стал первой криптовалютой, где применили блокчейн в качестве структуры данных. А при создании Ethereum наряду с решением для криптовалюты, похожим на Bitcoin, предложили использовать смарт-контракты, которые задействуют такие возможности блокчейна, как неизменность и принцип распределенного консенсуса. Идея смарт-контрактов впервые была предложена значительно раньше: Ник Сабо (Nick Szabo) сделал это еще в 1990-е. Согласно определению, это «набор обязательств, облеченных в цифровую форму, включая протоколы, в которых стороны действуют в соответствии с этими обязательствами». В системе Ethereum смарт-контракт – это программный код, который доставляется в сеть, так чтобы доступ к нему был у каждого. И результат выполнения этого кода удостоверяется механизмом согласования данных (консенсуса) и каждым участником сети в целом.

То, что мы сегодня называем блокчейном, — это совокупность нескольких технологий: самой по себе структуры данных блокчейна (цепочки блоков), алгоритмов распределенного консенсуса, криптографии с открытым ключом и смарт-контрактов. Ниже мы рассмотрим эти технологии более подробно.

Как уже отмечалось ранее, сам по себе термин «блокчейн» обозначает структуру организации данных. Все записанные данные разделены на блоки, где каждый блок данных содержит хеш всех данных предыдущего блока. Цель применения такой структуры данных в том, чтобы достичь доказуемой неизменности: если часть данных изменяется, необходимо пересчитать хеш того блока, который ее содержит, равно как и хеши всех последующих блоков. Это значит, что в качестве гарантии неизменности всех данных выступает только хеш самого последнего блока. В блокчейн-решениях данные, которые хранятся в блоках, формируются в результате всех транзакций, которые были подтверждены в процессе их создания. Это означает, что никто не может вставить, удалить или изменить транзакции в уже подтвержденном блоке, не оставив эти действия незамеченными. Начальный «нулевой» блок, который называется genesis-блоком, обычно хранит некоторые сетевые настройки, например публичные ключи валидаторов (тех, кто создает блоки).



Блокчейн-решения разрабатываются для использования в распределенной среде. Предполагается, что узлы сети хранят идентичные данные и образуют одноранговую сеть, где отсутствует центральный орган управления. Чтобы достичь согласия по данным в блокчейне, применяется алгоритм консенсуса, устойчивый к действиям злоумышленников. Это свойство консенсуса называется византийской отказоустойчивостью (BFT), свое название оно получило по аналогии с известной задачей византийских генералов. В блокчейн-решениях применяются различные алгоритмы, обладающие византийской отказоустойчивостью. Те, что предназначены для использования в полностью децентрализованных самоорганизующихся сетях, таких как платформы криптовалют, применяют алгоритмы доказательства выполнения работы (Proof of Work) или доказательства владения доли (Proof of Stake). При этом валидаторы выбираются алгоритмом таким образом, что им экономически выгодно действовать честно. В случае если система требует

определенного разрешения для валидации транзакций, валидаторы могут выбираться на этапе создания сети. Суть в том, что все валидаторы выполняют все входящие транзакции, и для того чтобы прийти к консенсусу по поводу результатов выполнения транзакций, необходимо согласие более двух третей валидаторов..

Использование криптографии с открытым ключом преследует главным образом две цели: во-первых, все валидаторы владеют своими собственными ключевыми парами, которые используются для подписи консенсус-сообщений, и во-вторых, все входящие транзакции (запросы на изменение данных блокчейна) должны быть подписаны, чтобы была возможность установить автора запроса. Анонимность в контексте блокчейн-технологий относится к тому обстоятельству, что каждый желающий пользоваться криптовалютой должен сгенерировать случайную ключевую пару и использовать ее, чтобы управлять кошельком, привязанным к открытому ключу. Блокчейн гарантирует, что управлять средствами в кошельке может только владелец ключевой пары, и это свойство доступно для проверки. Что же касается онлайн-голосования, бюллетени требуется принимать анонимно, но только от легитимных избирателей. Поэтому сам по себе блокчейн определенно не может решить проблему конфиденциальности голосования.

Второе дыхание блокчейн-решения обрели в лице смарт-контрактов. Они послужили стимулом для применения технологии блокчейн в попытках развития самых разных областей. Сам по себе смарт-контракт — это не более чем часть логики, записанной в виде кода. Однако в связке с неизменностью данных, которую предоставляют блокчейн и распределенный консенсус, смарт-контракт может выступать как безусловно доверенная третья сторона. Будучи однажды написанным, он уже не может быть изменен, и все выполняемые им действия верифицируются всеми участниками сети. Но поистине замечательное свойство смарт-контрактов — это то, что каждый, кто может развернуть узел блокчейн-сети, может и верифицировать результат выполнения смарт-контракта.

Несмотря на эти проблемы, системы онлайн-голосования по-прежнему могут выиграть от применения технологий блокчейна по двум основным причинам: неизменность данных и, в частности, проверка выполнения смарт-контрактов.

Как у любой другой технологии, у блокчейна есть свои недостатки. В отличие от других распределенных решений, масштабирование блокчейна действительно трудно осуществимо: растущее число узлов не улучшает производительность сети, потому что по определению каждый узел должен выполнять все транзакции, и этот процесс не разделяется между узлами. Более того, на производительность влияет рост числа валидаторов, так как это подразумевает более интенсивный обмен сообщениями в процессе достижения консенсуса. По той же причине блокчейн-решения уязвимы перед угрозой различных атак типа «отказ в обслуживании». Если блокчейн позволяет опубликовать смарт-контракт в сети любому желающему, то работу всей сети можно нарушить, всего лишь создав в смарт-контракте бесконечный цикл. Также сеть можно атаковать, просто отправив очень большой объем транзакций: в какой-то момент система откажется принимать что-либо еще. В криптовалютных решениях все транзакции имеют свою стоимость выполнения: чем больше ресурсов использует транзакция, тем дороже обходится ее исполнение. Кроме того, существует порог стоимости, и транзакции с комиссией ниже этого порога отклоняются. В приватных блокчейн-сетях эту проблему решают в зависимости от того, как сеть реализована на практике: посредством того же механизма комиссий, контроля доступа или иным способом, который больше подходит в конкретном случае.

Несмотря на упомянутые проблемы, использование технологии блокчейн в системах онлайн-голосования все равно имеет преимущества. На то есть две причины: неизменность данных и, в особенности, верификация выполнения смарт-контрактов.

Описание решения Polys

Введение

Решение Polys — это не просто блокчейн-платформа. Ключевые компоненты нашей системы:

- блокчейн-платформа
- сервисный слой
- библиотека polys-protocol

Наше решение для онлайн-голосования Polys основано на технологии блокчейн, чтобы обеспечить максимальную проверяемость на каждом этапе процесса голосования. Наша система позволяет проводить голосование в различных областях — ее могут использовать муниципальные власти, политические партии, студенческие организации, она подходит для инициативного бюджетирования и т. д. В зависимости от назначения различаются и требования, предъявляемые к системе, поэтому, чтобы наша система могла удовлетворять всем этим требованиям, мы сделали ее модульной. В ней представлены различные виды бюллетеней: с единичным и множественным выбором, с распределением кумулятивных голосов, бюллетени для референдума. В одних случаях не требуется анонимность, в других каждый голос участника голосования имеет определенный вес (к примеру, при голосовании акционеров). Мы предоставляем организатору голосования возможность выбрать необходимые свойства. Мы также готовы разработать иные возможности системы голосования, которые не представлены в рамках нашей базовой платформы.

Чтобы обеспечить модульность системы, мы делаем изолированными некоторые структурные элементы и процессы в домене, предназначенном для голосования. Это следующие структурные элементы:

- **Учетная запись (двух типов):** учетная запись организатора — чтобы инициировать процесс голосования и управлять им, и учетная запись участника голосования.
- **Список управления доступом (access control list, ACL):** список голосующих, имеющих право голоса. Списки управления доступом могут использоваться повторно, что обеспечивает организацию нескольких видов голосования в рамках одного общего события.
- **Голосование:** обладает такими общими характеристиками, как предмет голосования и доступные варианты выбора, даты начала и окончания, ACL, настройки управления голосованием, тип бюллетеня и способ подсчета.
- **Список участников:** обеспечивает контроль выдачи бюллетеней и процесс их принятия для конкретного голосования. К примеру, если применяется анонимизация с использованием слепой подписи, при выдаче бюллетеней слепая подпись сохраняется в качестве признака того, что избиратель получил свой бюллетень. А при принятии бюллетеней происходит проверка подписи, отправленной вместе с бюллетенем. Если анонимизация не требуется, то выдача бюллетеней пропускается, а наличие у участников права голоса проверяется на этапе принятия бюллетеней.
- **Хранилище бюллетеней:** определяет тип бюллетеней, используемых в голосовании. Зашифрованные или нет, с единичным выбором или множественным, с распределением кумулятивных голосов и т. д.
- **Хранилище результатов подсчета:** определяет способ подсчета голосов для подведения итогов голосования. Как правило, голоса избирателей просто суммируются, но есть также и такие способы подсчета, как в преференциальном (ранжированном) голосовании, когда процесс определения победителя не столь очевиден.

Не все комбинации структурных элементов допустимы. Но там где это возможно, мы разрешаем их комбинацию для удовлетворения потребностей пользователей системы.

Не все сочетания структурных элементов допустимы. Например, списки управления доступом с указанием веса голоса избирателей невозможно напрямую комбинировать с анонимизацией при помощи слепой подписи: такие списки требуют особого подхода. Или же преференциальное голосование лишено смысла в случае использования бюллетеней с единичным выбором. Но мы допускаем комбинацию структурных элементов, если таковая возможна, чтобы удовлетворить запросы пользователей системы.

Блокчейн-платформа — не единственный компонент решения Polys. Перечислим ключевые составляющие нашей системы.

- Блокчейн-платформа: предоставляет среду выполнения смарт-контрактов, определяющих логику процесса голосования.
- Сервисный слой: отвечает за аутентификацию пользователя, подпись ослепленных сообщений и решение инфраструктурных задач, таких как отправка уведомлений.
- Библиотека polys-protocol: обеспечивает слой абстракции над протоколом взаимодействия с блокчейном и позволяет оперировать высокоуровневыми структурными элементами (упомянутыми выше).

Дополнительные компоненты:

- **панель организатора**
- **панель участника голосования**
- **пакет программного обеспечения для наблюдателей**

Дополнительные компоненты также имеют важное значение:

- Панель организатора: разработана в виде веб-приложения, основанного на библиотеке polys-protocol, и обеспечивает графический интерфейс организатора голосования.
- Панель участника голосования: разработана в виде веб-приложения, основанного на библиотеке polys-protocol, и обеспечивает графический интерфейс участника голосования.
- Пакет программного обеспечения для наблюдателей. Состоит из трех частей: блокчейн-узел аудитора (полностью функциональный узел блокчейна, который не может создавать блоки), серверный компонент, который анализирует данные из блокчейна и обеспечивает программный интерфейс HTTP API для удобства извлечения информации, а также графический веб-интерфейс, отображающий эти данные.

Процесс генерации ключей происходит не в ключевых компонентах нашей системы: по умолчанию ключевая пара для шифрования бюллетеней генерируется в приложении организатора и хранится вместе с данными организатора.
По запросу мы предоставляем приложение для разделения ключа или сервис для распределенной генерации ключа.

Блокчейн-платформа

Как отмечалось выше, мы используем технологию блокчейн в качестве основы, чтобы обеспечить системе голосования проверяемость благодаря неизменности записанных в блокчейн данных и способности верифицировать логику всех процессов, определяемых смарт-контрактами. Наше решение построено на базе блокчейн-фреймворка Exonum. Далее мы приводим причины, почему выбрали именно его.

Фреймворк Exonum разработан преимущественно для организации блокчейн-сетей с эксклюзивным доступом (permissioned blockchain), которые наиболее подходят для онлайн-голосования. Мы не можем делать предположение о том, кто больше всего заинтересован в честном функционировании сети, как это происходит в случае публичного блокчейна. Равно как не можем ограничивать доступ к системе для всех, кроме организаторов голосования: такое действие исключило бы все преимущества блокчейна. Итак, следует создать несколько доверенных субъектов, уполномоченных участвовать в достижении консенсуса и создании блоков, в то время как другим участникам разрешено получать все данные из блокчейна. Чтобы исключить возможность сговора субъектов с целью переписать историю блокчейна, можно использовать механизм привязки (anchoring). Привязка – это периодическая публикация хеша самого последнего блока в публичную блокчейн-сеть вроде Bitcoin или Ethereum. В случае использования механизма привязки, даже если субъекты вступают в сговор и переписывают историю блокчейна, это можно легко обнаружить, потому что хеши переписанного блокчейна будут отличаться от хешей, которые будут хранится в публичном блокчейне.

Привязка — это процесс периодической публикации хэша последнего блока в публичную блокчейн-сеть, такую как Bitcoin или Ethereum.

Фреймворк Exonum написан на языке программирования Rust и позволяет разрабатывать на нем логику смарт-контрактов, что является отличным вариантом для создания надежных и быстрых приложений. Публичные блокчейны применяют различные способы, чтобы предотвратить выполнение логики, которая могла бы нарушить работу сети. В Ethereum, например, используется виртуальная машина Ethereum, которая предоставляет полный по Тьюрингу язык ассемблера (на его основе построен язык Solidity). При этом сама по себе виртуальная машина позволяет предсказывать сложность выполнения функции и исключить чересчур емкие операции. Это можно считать частным решением проблемы останова. В некоторых других блокчейнах для смарт-контрактов используются неполные по Тьюрингу языки, что по определению делает невозможным создание бесконечного цикла. У Exonum подобных ограничений нет. Это означает, что логику смарт-контрактов необходимо тщательно проверять, однако это компенсируется тем фактом, что смарт-контракты не может публиковать любой желающий: смарт-контракты могут быть введены в действие только валидаторами и лишь тогда, когда они достигнут консенсуса о его развертывании в сети. Это решение для создания смарт-контрактов показывает наиболее высокую производительность среди всех блокчейн-решений — до 5000 транзакций в секунду, а иногда и больше.

Exonum обладает инструментами, которые позволяют в режиме реального времени обеспечить всех участников сети надежными доказательствами того, что логика работы системы является корректной.

В фреймворке Exonum используется специальный алгоритм консенсуса на основе PBFT. Также в нем применяется ряд идей, реализованных в Tendermint, но с некоторыми отличиями. Как любой алгоритм, основанный на BFT, он обладает свойством финальности (finality), что исключает возможность возникновения форка — появления нескольких блоков на одной высоте цепочки. Для верификации блока требуется не просто подтверждение лидера текущего раунда, как в случае с алгоритмом Parity Aura: верификация возможна, только если будет получено согласие более двух третей валидаторов. Для работы Exonum не требуются майнинг или иные экономические стимулы.

Кроме того, Exonum обладает инструментами, которые позволяют каждому получить криптографическое доказательство существования в блокчейне определенных данных. А также того факта, что данные были добавлены в блокчейн в результате транзакции, одобренной существующими валидаторами. Эти инструменты позволяют в режиме реального времени обеспечить всех участников сети надежными доказательствами того, что логика работы системы является корректной. В том числе и клиентов с ограниченными вычислительными ресурсами (в особенности это относится к веб-приложениям организаторов и участников голосования). Все это обеспечивает высокую степень прозрачности системы.

Подробности процесса голосования

Как мы отметили ранее, наше решение способно адаптироваться к различным требованиям, и мы не станем пытаться охватить в этом обзоре все возможные варианты его использования. Вместо этого мы сделаем пошаговое описание процесса голосования, как он выглядит по умолчанию. Мы объясним, что и как можно проверить на каждом из этапов, и прокомментируем возможные модификации.

Этап 0. Первоначальное развертывание

Ядром системы является блокчейн-сеть. Чтобы развернуть ее, необходимо выбрать валидаторов. Каждый из них должен сгенерировать свою собственную ключевую пару для подписи сообщений, далее следует выполнить обмен открытыми ключами и добавить их в конфигурацию узла. Список валидаторов записывается в genesis-блоке базы данных блокчейна. Поэтому, однажды создав, его уже нельзя изменить никаким другим способом, кроме использования механизма консенсуса. Открытые ключи узлов должны находиться в общем доступе, чтобы существовала возможность проверки криптографических доказательств, предоставляемых блокчейн-платформой. Эти доказательства используются не только для ретроспективной проверки работы системы, но также и для того, чтобы убедиться, что пользователь обращается к честному узлу сети, а не к злоумышленнику, действующему по схеме man-in-the-middle. Если необходимо использовать механизм привязки, его следует настроить на этом этапе. По умолчанию платформа Exonum предусматривает привязку к сети Bitcoin, но можно легко реализовать привязку к любой другой системе.

Когда сеть настроена и запущена, необходимо выполнить конфигурацию сервисов. Наряду с информацией о версиях используемых сервисов в ней содержатся данные об открытых ключах учетных записей сервисного слоя. Это единственное место, где эти учетные записи уполномочены совершать операции. Эти операции связаны с представлением учетных записей пользователей в блокчейне. Дело в том, что решение Polys предназначается для использования с внешней системой идентификации. Поэтому сначала пользователь должен пройти в ней процедуру аутентификации. А затем отправить запрос к сервисному слою, чтобы увязать свою ключевую пару для подписи транзакции, сгенерированную случайным образом, со своей учетной записью в блокчейне. В сервисном слое хранится информация, необходимая для того, чтобы подтвердить аутентификацию пользователя с помощью внешней системы идентификации и установить соответствие внешнего идентификатора пользователя тому внутреннему идентификатору, который используется в блокчейне. Это означает, что конфиденциальная личная информация содержится только в сервисном слое и пользователю не приходится помнить ключевые пары или свои мнемонические фразы – их у него может быть неограниченное количество. Более подробная информация об этом процессе приведена в разделе «Этап 2. Аутентификация участника голосования». Процесс аутентификации организатора выглядит похожим образом.

Для сервисного слоя необходимо сгенерировать ключевую пару сервиса, взаимодействующего с блокчейном, а также выполнить настройку соединений базы данных или параметров почтового сервиса. Как замечено выше, владелец данной ключевой пары уполномочен связывать учетные записи внешней системы идентификации с учетными записями в блокчейне. Поэтому с данной ключевой парой следует обращаться осторожно. Кроме того, необходимо сгенерировать ключевую пару для сервиса, подписывающего сообщения вслепую. По умолчанию в Polys используется одна ключевая пара, но можно изменить эту настройку, чтобы использовать пороговую мультиподпись. Сервис подписи может быть отделен от других сервисов.

В приложениях организатора и участника голосования, как и в ПО наблюдателя, отсутствуют настройки, которые требовали бы особого отношения, — в них используется только публично доступная информация, такая как настройки интерфейсов блокчейна, открытые ключи валидаторов и т. д.

Этап 1. Подготовка голосования

Процесс подготовки голосования начинается с того, что определяется список управления доступом (ACL) с указанием идентификаторов участников, которые имеют право голоса. Параметры ACL следующие: способ аутентификации участника голосования, имеют ли голоса участников одинаковый вес или нет и определен ли список участников заранее (его предоставляют организаторы) или же участники могут сами добавлять себя в список, например в случае публичного голосования. В зависимости от того, какая система используется для идентификации (по умолчанию Polys обеспечивает идентификацию

по электронной почте, номеру телефона и через унифицированный модуль по протоколу OAuth 2.0), организатор подготавливает список соответствующих идентификаторов и создает запрос на создание ACL в адрес сервисного слоя. Сервис создания ACL присваивает каждому участнику голосования внутренний идентификатор и хранит привязку к внешнему идентификатору в базе данных сервиса. Сервис отправляет транзакцию создания ACL и регистрирует список внутренних идентификаторов. Отметим, что создавать списки ACL и добавлять в них записи разрешено только сервису создания ACL, потому что этот процесс предполагает доступ к конфиденциальным данным участников голосования. Так как список ACL создается в блокчейне, просматривать его может каждый желающий, однако обычным пользователям доступен лишь список внутренних идентификаторов и регистрационный статус. Для заранее установленных списков сервис также отправляет транзакцию, которая останавливает регистрацию, тем самым предотвращая добавление новых записей — даже со стороны сервиса. Если требуется провести аудит настоящих идентификаторов (например, наблюдателями), можно предоставить доступ к базе данных сервиса. Также существует вариант хранения хеша исходного списка (с внешними идентификаторами) в данных ACL в блокчейне в качестве гарантии неизменности списка.

Далее мы переходим к созданию собственно голосования. Организатор определяет идентификатор ACL, который будет использован, предмет голосования, перечень вариантов выбора, а также даты начала и окончания голосования. Также организатор выбирает способ участия в голосовании (на данный момент их два: неанонимный доступ и анонимный доступ с использованием слепой подписи), тип бюллетеня (с единичным выбором или множественным, зашифрованный или нет и т. д.) и метод подсчета. В данной работе мы рассмотрим установку по умолчанию, с использованием слепой подписи и зашифрованных бюллетеней.

Чтобы использовать зашифрованные бюллетени, необходимо сгенерировать ключевую пару для шифрования и сохранить в блокчейне открытый ключ. Здесь уместно упомянуть об используемых нами криптографических методах.

И для слепой подписи, и для шифрования бюллетеней применяется криптография на эллиптических кривых. Мы используем эллиптическую кривую Curve25519, а также **метод Ristretto**, чтобы обеспечить быстрые и надежные криптографические операции. Конкретно используемые нами реализации — **curve25519-dalek** для криптографических операций в смарт-контрактах и **JS-реализация ristretto255** для клиентских приложений (панелей организатора и участника голосования). Протокол слепой подписи мы подробно разберем далее, когда речь пойдет о получении и отправке электронного бюллетеня. Для шифрования бюллетеней мы используем схему Эль-Гамаля на эллиптических кривых (EC-ElGamal). Она является гомоморфной по сложению, что делает возможным вычисление суммы голосов до их расшифрования для ускорения процесса подсчета. Также она позволяет эффективно формировать доказательства принадлежности диапазону для верификации бюллетеней. Криптографический протокол построен на основе схемы, описанной в работе [1]. Подробнее мы рассмотрим этапы шифрования бюллетеней и подсчета голосов в разделах, посвященных процедурам отправки бюллетеня и подсчета голосов.

Генерация ключевой пары для шифрования бюллетеней — очень важный процесс. При этом необходимо обеспечить достаточную степень случайности, а доступ к секретному ключу со стороны злоумышленников должен быть абсолютно исключен. В действительности же далеко не всякий раз возникает необходимость выполнять этот процесс за закрытыми дверями без связи с внешним миром: разные голосования предполагают разные типы нарушителей. По умолчанию ради простоты и удобства использования в нашей системе ключевая пара генерируется в приложении организатора и хранится вместе с данными организатора. Мы предусмотрели вариант, при котором организатор может не хранить ключевую пару в системе и загрузить ее на свой компьютер или же сгенерировать ее вне нашего приложения и предоставить открытый ключ для создания голосования (а также секретный ключ непосредственно перед подсчетом голосов). По запросу мы предоставляем приложение для распределения ключей: разделения секрета по схеме Шамира или распределенной генерации ключей.

Помимо ключевой пары необходимыми параметрами являются базовая точка генерации эллиптической кривой и точки, составляющие выбор участника голосования. Для текущего набора типов бюллетеней выбраны две точки: одна показывает, что выбор сделан (точка выбора — *choicePoint*), а вторая — что выбор не сделан (пустая точка — *blankPoint*). Эти параметры жестко прописываются в блокчейне. Базовая точка генерации является частью параметров домена эллиптической кривой. Более подробно мы рассмотрим метод выбора точек в разделе, посвященном подсчету голосов.

Этап 2. Аутентификация участника голосования

В решении Polys используется внешняя система идентификации. По умолчанию мы предоставляем возможность идентификации по электронной почте, номеру телефона, а также с помощью модуля для использования любой системы, совместимой с протоколом OAuth 2.0. Кроме того, можно легко добавить какой-либо иной вариант. Вне зависимости от того, какая система идентификации используется, протокол аутентификации всегда один и тот же.

Пройдя аутентификацию, пользователи большей частью взаимодействуют с блокчейном напрямую. Для каждого пользователя в блокчейне создается сущность, называемая «псевдонимом» (Alias). Он содержит внутренний идентификатор пользователя и список принадлежащих ему открытых ключей. Информация о соответствии внешних идентификаторов внутренним хранится в базе данных сервиса аутентификации. Итак, на первом этапе аутентификации пользователь отправляет запрос сервису аутентификации, заявляя, что у него есть определенная внешняя учетная запись. В зависимости от способа аутентификации выбирается конкретный порядок действий. Например, при использовании электронной почты и телефона отправляется код подтверждения, а в случае аутентификации по протоколу OAuth 2.0 пользователь перенаправляется на страницу сервиса идентификации, где он должен выполнить вход. Когда пользователь подтверждает право владения внешней учетной записью, сервис аутентификации находит или создает соответствующий пользователю внутренний идентификатор и случайным образом генерирует некоторое секретное значение. Хеш этого секрета с внутренним идентификатором отправляется в блокчейн. Эту запись, состоящую из хеша секрета и внутреннего идентификатора, называют «запросом подтверждения». Собственно секрет отправляется обратно пользователю. Тот, в свою очередь, случайным образом генерирует ключевую пару для взаимодействия с блокчейном и подписывает транзакцию, в которой содержится полученный ранее секрет. Смарт-контракт проверяет секрет и, если хеши совпадают, связывает открытый ключ, полученный из подписи транзакции, с учетной записью, которая была определена в запросе подтверждения. Считается, что все дальнейшие транзакции, подписанные этой ключевой парой, отправлены этим конкретным пользователем.

Этап 3. Получение электронного бюллетеня

Чтобы обеспечить конфиденциальность участников голосования, мы используем схему слепой подписи. Эта схема описана в работе [2]. Схема обладает необходимыми свойствами неподложности (в предположении EC DLP; поставить подпись может только авторизованный субъект) и непрослеживаемости (подписывающий не может установить связь между ослепленными и неослепленными значениями). Протокол работает следующим образом.

На первом этапе участник голосования должен получить параметры домена слепой подписи — генератор эллиптической кривой G, открытый ключ подписи Q и случайную точку R ($R = k * G$; чтобы подписать ослепленное значение, подписывающий должен знать значение k). Ключевые пары хранятся в сервисе подписи, а их открытые составляющие публикуются в блокчейне, так что участник голосования получает параметры непосредственно оттуда.

Мы называем значение, подписываемое вслепую, токеном авторизации бюллетеня. Чтобы создать этот токен, участник голосования сначала должен сгенерировать случайную ключевую пару (не привязанную к его учетной записи) для взаимодействия с блокчейном, которая будет использоваться при подписании транзакции с бюллетенем. Токеном авторизации бюллетеня будет выступать хеш открытого ключа этой ключевой пары. Поскольку эта ключевая пара не связана с учетной записью участника голосования, она не раскрывает никакой конфиденциальной информации. А связь между открытым ключом и токеном авторизации бюллетеня дает возможность проверить, что тот, кто использовал токен, — это тот же самый человек, который владеет ключевой парой для подписания транзакции. Такой способ предотвращает атаки типа man-in-the-middle на канал передачи бюллетеней: даже если злоумышленнику удается перехватить транзакцию, он все равно не может воспользоваться токеном авторизации бюллетеня, чтобы отправить бюллетень. Потому что ему неизвестен секретный ключ для подписания транзакции, а смарт-контракт проверяет, чтобы хеш открытого ключа, выведенного из подписи, соответствовал токену авторизации.

На следующем этапе участник голосования выбирает три случайных значения в качестве факторов ослепления t_1, t_2, t_3 — и ослепляет токен авторизации бюллетеня (d — это секретный ключ подписывающего):

$$X = (t_1 R + t_2 G + t_3 Q) = (t_1 k + t_2 + t_3 d) G$$

$$m' = x_r t_1 (m^{-1} + t_3)^{-1}$$

где m' — ослепленное сообщение, а X будет отправлено как составная часть подписи.

На этом этапе все сгенерированные параметры — ключевая пара для подписи и факторы ослепления — могут быть сохранены в надежном хранилище. В случае прерывания процесса в результате сбоя сети или по иной причине это позволит участнику голосования восстановить процесс.

Ослепленное сообщение отправляется в сервис подписи. Этот запрос отправляет участник голосования, прошедший процедуру аутентификации. Сервис подписи первым делом проверяет, получил он подпись или еще нет. Эта проверка осуществляется логикой смарт-контракта в блокчейне: если сервис подписывает ослепленное значение для участника голосования, он сохраняет его вместе с подписью в блокчейне как признак того, что этот участник голосования получил подпись. И только после этого отправляет ее голосующему. Тем самым исключается возможность двойного голосования, потому что участник голосования никак не может получить подпись вторично. При последующих попытках участника голосования запросить подпись он получит ту, что хранится в блокчейне. Так что если процесс прервется, она не будет утрачена.

Подписание происходит так:
$s' = dx_r + km'$

Получив подпись для ослепленного токена авторизации, участник голосования использует факторы ослепления, чтобы создать подпись для неослепленного токена:
$s = m(t_1 s' m'^{-1} + t_2)$
подпись есть пара (X,s)

Подпись можно верифицировать, убедившись, что следующее равенство верно:
$sG = mX + Q$

Полученная в результате подпись вместе с токеном авторизации бюллетеня и самим бюллетенем отправляется по анонимному каналу. Отметим, что на самом деле для совершения этого процесса нет необходимости полагаться на наше приложение участника голосования. Имея все нужные параметры, избиратель может создать подпись для неослепленного токена при помощи любого другого программного обеспечения (либо специально написанного скрипта), которому он доверяет. А также он может сформировать и сам бюллетень и отправить транзакцию из любого места напрямую в блокчейн. Иными словами, после того как участник голосования получает подпись для ослепленного токена, процесс может быть прерван и без проблем возобновлен в любой момент и с любого другого устройства, без необходимости повторной аутентификации.

Этап 4. Отправка бюллетеня

Чтобы зашифровать свой выбор, участник голосования получает из блокчейна параметры домена эллиптической кривой, открытый ключ и точки, используемые для шифрования (точку выбора и пустую точку), а также список доступных вариантов. Способы создания бюллетеня немного отличаются в зависимости от типа бюллетеня. Мы рассмотрим бюллетени с единичным и множественным выбором, а также бюллетени для кумулятивного голосования (с распределением кумулятивных голосов):

1. Бюллетень с единичным выбором

В бюллетене этого типа голосующий может выбрать только один из представленных вариантов. По каждому пункту в списке голосующий зашифровывает точку: пустую точку в том случае, если пункт не выбран, и точку выбора для выбираемого им пункта:
$(A,B) = (r * G, blankPoint + r * Q)$
или
$(A,B) = (r * G, choicePoint + r * Q)$
где (A,B) — получившийся шифротекст, G — генератор эллиптической кривой, r — случайный параметр, генерируемый участником голосования, а Q — открытый ключ для шифрования.

Для каждой операции шифрования необходимо использовать разные сессионные ключи (случайные параметры). В противном случае для всех вариантов, которые не были выбраны, получится идентичный результат шифрования.

Затем необходимо сформировать доказательства принадлежности диапазону, чтобы подтвердить, что результирующий набор содержит только один выбор. Механизм создания доказательств работает следующим образом.

Доказывающий должен доказать, что выбор $C_p = (A_p, B_p)$ принадлежит множеству $\{M_1, M_2, ..., M_m\}$.
Доказывающий генерирует точки A_k, B_k, for $1 <= k <= m$:
$A_k = w_k * G + u_k * A_p, \ \forall k \neq c_p$
$A_{cp} = s * G$
$B_k = w_k * Q + u_k * (B_p - M_k), \ \forall k \neq c_p$
$B_{cp} = s * Q$
$challenge = hash(A_1, A_2, ... A_m, B_1, B_2, ... B_m),$
$u_{cp} = challenge - \sum_{k \neq cp} u_k,$
$w_{cp} = s - u_{cp} r_p$
где w_k, u_k, s — это случайные значения из группы. Доказывающий отправляет значения A_k, B_k, u_k, w_k проверяющему. Проверяющий удостоверяется в том, что:
$A_k = w_k * G + u_k * A_p, \ \forall k \in [1,m]$
$B_k = w_k * Q + u_k * (B_p - M_k), \ \forall k \in [1,m]$
$challenge = \sum_{k=1}^{m} u_k$
при этом *challenge* вычисляется как $hash(A_1, A_2, ... A_m, B_1, B_2, ... B_m)$

В случае бюллетеня с единичным выбором нам требуется доказать, что сумма всех зашифрованных вариантов в точности составляет *choicePoint + (n - 1) * blankPoint*, где *n* — общее число вариантов выбора.

2. Бюллетень со множественным выбором

В бюллетене этого типа участник голосования может выбрать несколько из предложенных вариантов — максимальное и минимальное значения определяются на этапе создания голосования и хранятся в блокчейне. Шифрование выполняется точно так же, как в случае бюллетеня с единичным выбором, но с учетом того, что можно выбирать несколько вариантов. А в отношении доказательств с нулевым разглашением дело обстоит несколько сложнее.

Во-первых, участник голосования должен доказать, что он выбрал количество вариантов не менее минимально необходимого и не более максимально установленного. Следовательно, допустимый набор доказательств с нулевым разглашением будет таким (*cP* обозначает *choicePoint*, а *bP* — *blankPoint*, *min* и *max* — соответственно минимальное и максимальное количество разрешенных к выбору вариантов):
*{min * cP + (max - min) * bP, (min + 1) * cP + (max - min - 1) * bP,...,max * cP}*

Однако этого недостаточно: так как голосующий имеет возможность выбрать более одного варианта, теоретически он может создать бюллетень как *max * cP* для одного варианта, зашифровав для всех остальных вариантов пустые точки. Подобные действия определенно являются нежелательными, поэтому для каждого зашифрованного варианта нам следует создать доказательства с утверждением, что в нем зашифровано только значение *cP* или *bP*.

3. Бюллетень для кумулятивного голосования

В бюллетене этого типа участник голосования должен распределить голоса между доступными вариантами. По своему усмотрению он может подать всю сумму голосов за один вариант либо разделить ее между несколькими вариантами. Минимальное и максимальное допустимое значение определяют на этапе создания голосования. По каждому варианту участник голосования создает значение для шифрования следующим образом:
*scores * cP + (max - scores) * bP*
где *scores* — количество голосов в пользу этого варианта. Смысл подобного шифрования каждого варианта в том, что нам необходимо получить заданное «число» зашифрованных голосов, — сумма множителей точки выбора и пустой точки должна быть равна максимальному числу голосов, которое имеет в своем распоряжении участник голосования. Причину этого мы подробно изложим в разделе, посвященном подсчету голосов.

Для доказательства с нулевым разглашением для суммы диапазон определяется таким образом (*n* — общее число вариантов):
*{min*cP +(max*n-min)*bP, (min+1)*cP +(max*n-min-1) *bP,...,max*cP +(max*n-max)*bP}*

Помимо проверки суммы необходимо проверить, что шифрование каждого варианта содержит число голосов от нуля до максимально установленного значения:
*{max * bP,cP + (max - 1) * bP, 2 * cP + (max - 2) * bP,...,max * cP}*

Получаемый в результате бюллетень создается на основе списка зашифрованных значений для каждого варианта, доказательства с нулевым разглашением для суммы зашифрованных значений, списка доказательств с нулевым разглашением для каждого варианта (если необходимо) и токена авторизации бюллетеня с подписью, которая была получена на предыдущем этапе. Все эти данные упаковываются в транзакцию и отправляются напрямую в блокчейн по анонимному каналу.

Чтобы найти транзакцию в блокчейне, участнику голосования достаточно иметь хеш транзакции, который он легко может вычислить. Блокчейн Exonum предоставляет криптографическое доказательство того факта, что транзакция прошла процедуру консенсуса всех существующих валидаторов. Это доказательство можно использовать в случае каких-либо спорных ситуаций. Также хеш используется как внутренний идентификатор бюллетеня, и блокчейн-платформа предоставляет возможность получить о нем всю информацию: как хранятся данные и результаты проверки. По окончании голосования его участник может использовать опубликованный секретный ключ и расшифровать бюллетень, чтобы проверить, содержится ли в бюллетене выбранный им вариант.

Этап 5. Подсчет голосов

Чтобы приступить к подсчету голосов, сначала необходимо остановить процедуру голосования и запретить поступление новых бюллетеней. Затем организаторы публикуют в блокчейне ключ расшифрования, и начинается соответствующий процесс.

Выполняется проверка токенов авторизации каждого бюллетеня и доказательств с нулевым разглашением. Если все правильно, зашифрованные значения каждого варианта поочередно суммируются с теми, что уже записаны: это означает, что по каждому варианту фиксируется отдельная сумма и каждое значение из зашифрованного списка вариантов в бюллетене прибавляется к соответствующей сумме.

Значения в этой сумме такие: $a*cP + x*bP$, где a — количество выбранных вариантов, а x — количество пустых точек. Чтобы получить количество выбранных вариантов, нужно решить задачу дискретного логарифмирования в ограниченном контексте. В работе [1] предлагается решение этой задачи как задачи о рюкзаке с использованием метода «встречи посередине» (meet-in-the-middle) с заранее вычисленной таблицей. Благодаря выбранному способу шифрования вариантов выбора мы можем использовать немного измененный подход.
В случае использования бюллетеней с единичным выбором сумма, получаемая в результате, будет
$a*cP + (n-a) *bP$, где n — число сохраненных бюллетеней. То есть чтобы вычислить значение a, мы можем просто рассчитать таблицу со всеми возможными вариантами значений:
$\{n * bP,cP + (n - 1) * bP, (2 * cP) + (n - 2) * bP,...,n * cP\}$
и выполнить поиск суммы в этой таблице, чтобы определить количество вариантов выбора. В случае крупномасштабных выборных кампаний на вычисление подобной таблицы может потребоваться очень много времени, однако мы можем ускорить процесс, прибегнув к предвычислению. Для этого будем «упаковывать» бюллетени в т.н. пакеты. Поскольку нам неизвестно число участников голосования, от которых поступят бюллетени, мы определим размер такого пакета, например в 100 000, и подготовим предвычисленную таблицу для этого количества. В ходе голосования этот пакет постепенно заполняется, и если он достигает определенного заранее значения, то формируется новый пакет с бюллетенями. Чтобы получить результаты, все пакеты с бюллетенями расшифровывают и суммируют значения, которые представляют число вариантов, выбранных голосующими в каждом отдельном пакете.

Если же количество поступивших бюллетеней меньше предварительно оцененного объема, то в подготовленной таблице нас интересуют значения, где вместо n используется именно это количество. Если предвычислить все возможные варианты, размер результирующей таблицы будет $n + (n - 1) + (n - 2) + ... + 1$, и он стремительно увеличивается по мере роста значения n, поэтому эффективность такого решения сомнительна. Вместо этого в случае неполных пакетов с бюллетенями мы можем применить дополнение, зашифровав необходимое количество пустых точек, равное $(n - s) * bP$, где s обозначает размер неполного пакета. Этот прием позволяет вычислить таблицу размера n.

В случае бюллетеня для кумулятивного голосования каждый вариант может получить некоторое «число» точек, определяемое максимальным количеством голосов, которым может распорядиться голосующий. Итак, если размер пакета равен n, то при кумулятивном голосовании этот пакет может содержать *n⁄ maxScores* бюллетеней. Если в результате деления *n* на *maxScores* получается остаток, то к каждому заполненному пакету следует применить дополнение *remainder * bP*.

Выбор пустой точки и точки выбора зависит от заранее определенного размера пакета. В качестве точки выбора служит базовая точка кривой *P*, а пустая точка выбирается как *(n + 1) * P*, чтобы любое количество точек выбора не могло рассматриваться как одна пустая точка. Поскольку точки выбора и пустые точки, также как и размер пакета, одинаковы для всех голосований, мы предвычисляем таблицу для расшифрования при компиляции кода узлов блокчейна.

Благодаря тому что ключ расшифрования публикуется в блокчейне, каждый может проверить весь процесс поэтапно, используя соответствующие инструменты, и удостовериться в правильности результатов.

Примечание о противодействии принуждению

В обзоре протокола мы не упомянули о противодействии принуждению, потому что такие инструменты не встроены в ядро системы. При разработке протокола приоритет отдается максимальной проверяемости, и, как мы уже отмечали, методики реализации противодействия принуждению непременно влекут за собой снижение уровня проверяемости. Тем не менее существует возможность изменить некоторые составляющие протокола, чтобы в определенной мере достичь противодействия принуждению. К примеру, секретный ключ не обязательно публиковать в блокчейне, если исходить из того, что расшифрование выполняется владельцами ключа и в блокчейне публикуется совокупный результат расшифрования с доказательствами корректности этого процесса. Это происходит во взаимодействии с доверенным средством (сервисом или устройством), который скрывает фактор случайности от участника голосования. Принуждающий не будет знать, что именно зашифровано. Однако существуют другие уязвимые места, такие как аутентификация и процесс выдачи бюллетеней.

Примечание об инструментарии для наблюдения за голосованием

Прямое взаимодействие пользователей с блокчейном и криптографическое доказательство обработки транзакций позволяют избирателю быстро удостовериться, что его транзакция с голосом не будет потеряна. Смарт-контракты в соответствии со своим предназначением обеспечивают проверяемое выполнение логики. А механизм привязки еще больше укрепляет уверенность в неизменности данных. Более того, криптографические доказательства можно использовать для подтверждения определенных действий в случае возникновения спорной ситуации. Как следует из протокола, за каждым этапом процесса можно наблюдать, обращаясь напрямую в блокчейн. Мы упомянули о нашем программном обеспечении для наблюдателей, которое предоставляет удобный пользовательский интерфейс для наблюдения за состоянием системы, а также API-интерфейс для автоматической обработки данных. Мы продолжим развивать функционал приложения для наблюдателей, а также инструменты, призванные удостоверить результат выполнения сложных процессов, таких как расшифрование бюллетеней.

Заключение

Конечно, мы не собираемся останавливаться на этом — еще предстоит проделать немалую работу. Мы постоянно исследуем новые способы улучшения протокола. Мы стремимся создать надежное и поддающееся абсолютной проверке решение для голосования, которое подходило бы для всех вариантов голосования.

Источники и ресурсы

[1] M.A. Cervero, V. Mateu, J.M. Miret, F. Sebe', J. Valera: "An Elliptic Curve Based Homomorphic Remote Voting System" (2014)
[2] Hossein Hosseini, Behnam Bahrak and Farzad Hessar: "A GOST-like Blind Signature Scheme Based on Elliptic Curve Discrete Logarithm Problem" (2013)

Cyber Threats News: www.securelist.com
IT Security News: business.kaspersky.com/

www.kaspersky.ru

kaspersky АКТИВИРУЙ БУДУЩЕЕ

GBD-7262 Q1/21 V1