Наталья И. Касперская[1], Василий В. Кузьменко[2], Рустем Н. Хайретдинов[3],
Андрей Ю.Щербаков[4]
[1, 2, 3] *Группа компаний InfoWatch,*
Верейская ул., 29, стр. 134, г. Москва, 121357, Россия
[1]*Московский институт электроники и математики им. А.Н.Тихонова НИУ ВШЭ*
Таллинская ул., 34, г. Москва, 123458, Россия
[2]*ООО КБ «Нэклис-Банк»*
Никитская Б. ул., 17, стр. 2, г. Москва, 125009, Россия
[4]*Центр развития криптовалют и цифровых финансовых активов ВИНИТИ*
Усиевича ул., 20., г. Москва, А-190, 125190, Россия
[1]*e-mail: natalya.kaspersky@infowatch.com, https://orcid.org/0000-0002-5205-679X*
[2]*e-mail: vasily.kuzmenko@infowatch.com, https://orcid.org/0000-0002-5042-2012*
[3]*e-mail: rustem.khairetdinov@infowatch.com, https://orcid.org/0000-0002-3391-7646*
[4]*e-mail: a.shcherbakov@c3da.org, https://orcid.org/0000-0002-1593-6704*

О ПОДХОДАХ К СОЗДАНИЮ УНИВЕРСАЛЬНОГО ДОВЕРЕННОГО РАСПРЕДЕЛЕННОГО РЕЕСТРА, ОБЕСПЕЧИВАЮЩЕГО НЕРАЗГЛАШЕНИЕ ДАННЫХ О СИСТЕМЕ

DOI: http://dx.doi.org/10.26583/bit.2019.2.07

Аннотация. В настоящее время в условиях действующих международных санкций весьма актуальна задача построения системы распределенных реестров, обладающих высоким транзакционным быстродействием и использующих симметричные криптографические алгоритмы, а также обеспечивающих минимальное или регулируемое разглашение данных об архитектуре системы и проводимых в ней транзакциях. Для решения этой задачи сформулировано понятие технологий «информационного чёрного ящика» для системы, утройство которой таково, что внешний наблюдатель не имеет никакой информации об устройстве системы, ее участниках и данных, хранимых и обрабатываемых в ней, и технологию «информационного серого ящика» для системы, в которой фиксируется состав информации, передаваемой во внешние системы, остальная же информация относится к неразглашаемой. На основе системно-аналитического подхода, формулирования модели нарушителя и выделения существенных требований к информационной системе, содержащей распределенный реестр, в статье решена проблема создания универсального доверенного распределенного реестра, устойчивого к внешним атакам, включая трекинг транзакций и обеспечивающего неразглашение данных об архитектуре реестра, приведен протокол помещения информации в распределенный реестр, получения информации из него и управления правами доступа, рассмотрен также пример практической реализации описанного решения. Сформулированная концепция создания доверенного защищенного распределенного реестра может являться методологической основой для формулирования ведомственных или национальных регулирующих требований в области цифровой экономики, а также послужить технической основой для разработки конкретных проектов в области защищенных систем, использующих распределенные реестры в сфере государственного управления, финансов и учетно-сервисных систем.

Ключевые слова: блокчейн, распределенные реестры, транзакция, трекинг, безопасность, нулевое разглашение.

Для цитирования: КАСПЕРСКАЯ, Наталья И. et al. О ПОДХОДАХ К СОЗДАНИЮ УНИВЕРСАЛЬНОГО ДОВЕРЕННОГО РАСПРЕДЕЛЕННОГО РЕЕСТРА, ОБЕСПЕЧИВАЮЩЕГО НЕРАЗГЛАШЕНИЕ ДАННЫХ О СИСТЕМЕ. Безопасность информационных технологий, [S.l.], v. 26, n. 2, p. 95-108, 2019. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/1202>. Дата доступа: 30 may 2019. doi:http://dx.doi.org/10.26583/bit.2019.2.07.

Natalya I. Kasperskaya[1], Vasily V. Kuzmenko[2], Rustem N. Khairetdinov[3],
Andrey Yu. Shcherbakov[4]
[1, 2, 3] *Group of companies InfoWatch,*
29 Vereyskaya str., building 134, Moscow, 121357, Russia
[2]*Moscow Institute of electronics and mathematics. A. N. Tikhonov HSE*
Tallinn str., 34, Moscow, 123458, Russia
[2]*Bank "Neklis-Bank"*
Nikita Big str., building 17, Moscow, 125009, Russia
[4] *Center for development of cryptocurrency and digital of financial assets VINITI*
Usievich str., 20, Moscow, A-190, 125190, Russia
[1]*e-mail: natalya.kaspersky@infowatch.com, https://orcid.org/0000-0002-5205-679X*
[2]*e-mail: vasily.kuzmenko@infowatch.com, https://orcid.org/0000-0002-5042-2012*
[3]*e-mail: rustem.khairetdinov @infowatch.com, https://orcid.org/0000-0002-3391-7646*
[4]*e-mail: a.shcherbakov@c3da.org, https://orcid.org/0000-0002-1593-6704*

About approaches to a universal distributed trusted registry to ensure confidentiality of system information

DOI: http://dx.doi.org/10.26583/bit.2019.2.07

Abstract. Currently, under the current international sanctions, the task of building a system of distributed registries with high transactional speed and using symmetric cryptographic algorithms, as well as providing minimal or controlled disclosure of data about the architecture of the system and the transactions carried out in it, is very important. To solve this problem, the concept of "information black box" technologies for the system is formulated, the triplicity of which is such that the external observer has no information about the structure of the system, its participants and the data stored and processed in it, and the technology of "information gray box" for the system, in which the composition of information transmitted to external systems is fixed, the rest of the information refers to the undisclosed. Based on the system-analytical approach, the formulation of the model of the offender and the allocation of essential requirements for the information system containing the distributed registry, the article solves the problem of creating a universal trusted distributed registry, resistant to external attacks, including transaction tracking and ensuring non-disclosure of data on the architecture of the registry, the protocol of placing information in a distributed registry, obtaining information from distributed registry and access rights management, also considered an example of practical implementation of the described solution. The formulated concept of creating a trusted secure distributed register can be a methodological basis for the formulation of departmental or national regulatory requirements in the field of digital economy, as well as serve as a technical basis for the development of specific projects in the field of secure systems using distributed registers in the field of public administration, finance and accounting and service systems.

Keywords: blockchain, distributed registries, transaction, tracking, security, zero disclosure.
For citation: KASPERSKAYA, Natalya I. et al. About approaches to a universal distributed trusted registry to ensure confidentiality of system information. IT Security (Russia), [S.l.], v. 26, n. 2, p. 95-108, 2019. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/1202>. Date accessed: 30 may 2019. doi:http://dx.doi.org/10.26583/bit.2019.2.07.

Введение

В последние годы в области построения доверенных и защищенных информационных систем большой интерес вызывает использование технологии распределенных реестров (блокчейн) в целях решения широкого круга задач, начиная от организации децентрализованных платежных систем и закачивая задачами интеграции цифровых платформ и объединения разнородных баз данных [1].

Необходимо заметить, что методология создания доверенных и защищенных информационных систем на основе распределенных реестров испытывает определенные

теоретические трудности. Это связано с тем, что технология блокчейн развивалась от практических потребностей путем «проб и ошибок».

В настоящее время существует ряд действующих проектов, в первую очередь блокчейн криптовалюты Bitcoin, блокчейн Etherium, который попытался исправить некоторые недостатки проекта Bitcoin [2], можно отметить также попытки синтезировать «конструкторы» распределенных реестров – проекты типа Hyperledger Fabric.

У всех этих проектов имеется ряд принципиальных недостатков, вызванных «болезнями роста» – когда информационная технология растет и развивается от практики. Аналогом процесса является развитие теории вероятностей от практических навыков по анализу статистики событий до непротиворечивой аксиоматики А.Н. Колмогорова в области вероятностной меры. Несмотря на то, что в области информатики никогда невозможно будет отказаться от движения «от практики», теоретическое осмысление вопроса безопасности распределенных реестров тем не менее настоятельно необходимо.

1. Теоретические проблемы существующих распределенных реестров

Первичным теоретическим заблуждением проектов распределенных реестров является тезис о том, что информационная система на основе распределенного реестра может быть первично построена «равноправно», «независимо» и «децентрализовано».

Под «равноправностью» понимается, как правило, равноправие участников, некоторая их одинаковость с точки зрения собственных возможностей, изначального недоверия друг к другу, также с точки зрения модели угроз той системы, в которую интегрирована технология распределенного реестра.

Независимость системы декларируется с точки зрения невлияния некоторых «регуляторов» на нее и децентрализованность с точки зрения распределенности ресурсов в первую очередь, их доступности.

Однако, очевидно, что полная равноправность невозможна в первую очередь из-за того, что у практической реализации, по меньшей мере, части компонентов системы имеется «автор», который априорно имеет больше знаний о системе и возможностях по ее доработке и изменению [3].

Далее, у системы имеется условный «оператор», а практически – владелец, который разворачивает и поддерживает технические средства информационной системы, в которую интегрирована технология распределенного реестра. Владелец или автор практически единолично вносят изменения-форки (fork), которые могут полностью изменить систему, а с другой стороны – держат ее под контролем владельца. К этой же области относится сопровождение программного обеспечения распределенного реестра и исправление ошибок в нем.

Кроме того, за скобки выносится телекоммуникационная часть системы, которая доносит информацию клиентов (пользователей) до оператора (операторов) распределенных реестров. В современном мире именно телекоммуникации являются инструментом контроля и ограничения децентралированности и вполне понятно – что построить полностью независимую информационно-телекоммуникационную систему – значит полностью продублировать каналы связи, обеспечивающие передачу информации в ней, что является возможным только для проектов глобального уровня.

Таким образом, «равноправие», «независимость» и «децентрализованость» являются мифами, которые неосознанно или сознательно распространяются блокчейн-сообществом [4].

Надо обратить внимание, что в основу «доверия» к системам распределенных реестров положены криптографические задачи, например, задача построении коллизий хеш-функции или ассиметричные криптографические алгоритмы.

Задача доверенного обмена открытыми ключами для обеспечения корректного их использования, например, для обеспечения переводов с одного кошелька на другой решается только при помощи сертификатов – подписания открытого ключа в доверенном центре. А наличие доверенного центра перечеркивает децентрализованность полностью в первую очередь с точки зрения доступности, поскольку удостоверяющий центр может полностью регулировать выдачу сертификатов открытых ключей.

С другой стороны, «багаж» ассиметричной криптографии не позволяет выстроить быстродействующие и легко управляемые системы распределенных реестров в первую очередь из-за невысокой скорости ассиметричных криптографических алгоритмов. Кроме того, в умах неспециалистов ассиметричная криптография является некой панацеей для решения всех задач, что создает исходно ложные парадигмы проектирования защищенных систем.

Другой важной проблемой является тезис о том, что незамкнутая система не может быть защищенной с точки зрения формальной доказательности этого факта [5]. Исходя из этого, пользователи системы должны быть поименованы, а с другой – являться анонимными относительно друг друга. Кроме того, в системе в обязательном порядке должны присутствовать механизмы (с точки зрения системно-аналитических моделей компьютерных систем – субъекты) разграничения доступа, а в общем случае – реализации произвольно заданной политики безопасности.

В этой связи уместно привести пример неработоспособности семейства ассиметричных криптографических алгоритмов для решения задач анонимизации. Один из известных принципов построения анонимного имени для субъекта или объекта распределенного реестра – вычисление хеш-функции от открытых данных (имени или паспортных данных пользователя). Легко видеть, что перебор исходной информации (например, по базам данных паспортов) позволяет с невысокой трудоемкостью найти реальное имя по значению хеш-кода.

Заметим также, что формулирование как архитектурных, так и технических решений в области построения защищенных компьютерных систем находится в настоящее время и в политической плоскости, в первую очередь из-за наличия международных санкций в области экономики и финансов.

При этом возникает интересная ситуация – решения в области безопасности (это касается не только распределенных реестров) должны обеспечивать защиту от внешнего нарушителя высокого уровня (уровня развитой зарубежной спецслужбы типа АНБ США), включая требования получения «нулевых» знаний об участниках системы и передаваемой ими информации, а с другой стороны – наличие национального регулирования делает необходимым идентификацию участников и раскрытие в определенных законами и регламентами случаях информации о них и переданной между ними информации.

Наконец – интеграция систем распределенных реестров (особенно в области финансовых технологий) в открытые международные системы (например, передача транзакции из корпоративной финансовой системы в сеть Etherium или Bitcoin) делает необходимой реализацию «близких к нулю» знаний об участниках.

Приведем пример последнего требования. Например, транзакция, характеризующаяся отправителем, получателем и суммой, может отображаться во внешний распределенный реестр в виде изменяющихся каждый раз имен отправителя и получателя и открытой суммы, которая может быть нужна для выполнения внутренних требований в системе, например, ограничения суммы или для контроля балансов.

Следовательно, помимо технологий «информационного чёрного ящика» для системы, утройство которой таково, что внешний наблюдатель не имеет никакой

информации об устройстве системы, ее участниках и данных, хранимых и обрабатываемых в ней, можно предложить и технологию «информационного серого ящика» для системы, в которой фиксируется состав информации, передаваемой во внешние системы, остальная информация относится к неразглашаемой.

Эти положения являются принципиально новыми для теории компьютерной безопасности и позволяют формулировать новые задачи и результаты, касающиеся сопряжения различных систем.

2. Основные требования к защищенной информационно-телекомуникационной системе, включающей распределенный реестр

Первоначально опишем элементы рассматриваемой системы. Итак, рассматриваем множество пользователей, передающих информацию в виде законченных блоков (единиц) друг другу через систему хранения данных, представляющих собой распределенный реестр. Далее единицы этой информации будем также именовать транзакциями. Существует внешний наблюдатель (нарушитель), который получает информацию о системе, не являясь ее легальным пользователем. Ситуация, когда нарушитель контролирует одного или нескольких легальных пользователей, нуждается в отдельном рассмотрении. В некоторых случаях нарушитель может искажать единицы информации и блокировать их (уничтожать их в канале связи или телекоммуникационной среде). Важным является то, что транзакции характеризуются их отправителем и получателем (получателями), т.е. в составе транзакции имеются имена отправителя и получателя. Требование нулевого разглашения информации о системе определяет необходимость того, что для внешнего наблюдателя имена отправителя и получателя должны быть неизвестны либо каждую транзакцию изменяться. Это факт легко доказуем от противного, поскольку иначе нарушитель может определить число пользователей в системе и связи между ними, то есть требование нулевого разглашения оказывается невыполненным.

С учетом архитектуры и сформулированного принципа нулевого разглашения для внешнего наблюдателя система распределенных реестров должна в обязательном порядке обеспечивать:

– формирование приватного элемента для пользователя с гарантированными вероятностными свойствами, т.е. пользователь должен иметь приватный идентификатор или ключ, никому не известный кроме него, выработанный при помощи датчика случайных чисел с гарантированными статистическими свойствами;

– формирование сетевого имени (идентификатора, которым пользователь представляется в системе) на основе указанного выше приватного элемента, исключающего возможность выявления связей (со стороны внешнего наблюдателя) между сетевым именем и множеством данных о физическом лице или организации (далее – информация о пользователе), с другой стороны, сетевое имя должно быть проверяемым, т.е. уполномоченный орган должен при помощи детерминированной процедуры иметь возможность убедиться в соответствии сетевого имени и информации о пользователе;

– безопасное хранение приватного элемента у пользователя для обеспечения защищенности от несанкционированного доступа к нему;

– наличие «точки входа» для пользователей – оператора распределенного реестра, который на основе заданных государственным или ведомственным регулятором регламентов обеспечивает обработку информации пользователей;

– авторизацию пользователя для оператора при помощи криптографических процедур, использующих приватный элемент пользователя (далее он назван сетевым

ключом пользователя), известный также и оператору, но неизвестный другим пользователям;

– безопасный транспорт (как минимум с сохранением неизменности информации (транзакции), получаемой от пользователя) для передачи информации от пользователя к оператору распределенного реестра;

– контроль целостности и авторства каждой информационной единицы, помещаемой в распределенный реестр;

– формирование подтверждений у оператора распределенного реестра о факте помещения информации в распределенный реестр (например, путем выдачи заверенных оператором квитанций пользователям);

– наличие механизма формирования и обработки запросов по выдаче информации из распределенного реестра по запросам его участников (клиентов), обеспечивающего защищенность данного запроса (также авторизацию и контроль неизменности запроса);

– реализацию у оператора распределенного реестра системы разграничения доступа к информации (к звеньям распределенного реестра) в распределенном реестре;

– наличие механизма обеспечения «нулевых знаний» о структуре системы, ее участниках и транзакциях посредством процедуры изменения имен отправителя и получателя, а также других данных транзакции, зависящей от сетевого ключа пользователя.

Сделаем важное замечание – оператор распределенного реестра является доверенным элементом и знает имена пользователей, поскольку он должен выполнять между ними разного рода действия, например, в общем случае передавать информацию либо изменять состояние счетов пользователей, если транзакции носят финансовый характер.

Необходимо также уточнить модель нарушителя, которая неявно используется при формировании указанных свойств. В данном случае речь идет о модели внешнего нарушителя (модель H_2) – нарушителя который может читать и изменять информацию в каналах связи (в телекоммуникационной компоненте информационно-телекоммуникационной системы). Полагаем, что владелец системы (оператор распределенного реестра) является доверенным лицом или организацией, заинтересованной в корректной и безопасной работе всей системы. Как показано выше, это предположение является базовым для построения всей системы.

Краткое описание протокола защищенного доверенного распределенного реестра

Введем следующие обозначения:

X_i – пользователь распределенного реестра;

A_i – цифровая информация, описывающая пользователя РР (фамилия и имя, паспортные данные);

K_{pi} – персональный ключ (персональная информация) пользователя РР;

K_{si} – сетевой ключ пользователя (также являющийся частью персональной информации пользователя), предназначенный для связи с оператором РР;

C_i – ключевой контейнер пользователя, представляющий собой персональную информацию пользователя (персональный или сетевой ключ), закрытый на пароле пользователя при помощи обратимой криптографической процедуры;

S_i – сетевое имя пользователя, однозначно связанное с A_i;

$INFO_{ij}$ – информация i-го пользователя, сформированная на рабочем месте пользователя и направляемая для хранения и обработки в РР, имеющая условный номер j;

KV_{ij} – квитанция, сообщающая о результате обработки j-го информационного блока для i-го пользователя;

V_m – запрос на извлечение информации из РР;

K_o – ключ оператора, служащий для заверения цепочки данных в РР;

$I = Im(x, k)$ – функция вычисления имитовставки от информации x на ключе k;

$y = E(x, k)$ – функция зашифрования информации x на ключе k;

и

$x = D(y, k)$ – обратная операция – функция расшифрования информации y на ключе k.

В данном случае рассматриваем симметричные криптографические алгоритмы, когда для зашифрования и расшифрования используется один и тот же ключ.

Можно видеть, что функция вычисления имитовставки обладает возможностью как авторизации пользователя, так и контроля целостности передаваемой и хранимой информации. В связи с этим будем называть функцию вычисления и проверки имитовставки кодом аутентификации (КА).

Для обеспечения информационного взаимодействия пользователей и оператора необходимо обеспечить функционирование сервера оператора (сервер приема-выдачи информации распределенного реестра), имеющего следующие области для передачи данных:

– область приема данных сервера, в которую пользователи передают данные для помещения в распределенный реестр и запросы для выгрузки данных из распределенного реестра;

– область данных, в которую перемещаются объекты ошибочного формата (например, не имеющие кода аутентификации пользователя);

– область данных, содержащая квитанции о помещении информации в распределенный реестр;

– область данных выгрузки данных по запросам пользователей.

Полагаем, что пользователь системы имеет персональный вычислитель (ноутбук, смартфон или выделенный криптокомпьютер), подключенный при помощи каналов связи (телекоммуникационной среды) к серверу приема-выдачи информации РР.

Для регистрации в системе пользователь X_i при помощи датчика случайных чисел с гарантированными статистическим свойствами создает ключи K_{pi} – персональный ключ (персональная информация) пользователя РР и K_{si} – сетевой ключ пользователя (также являющийся частью персональной информации пользователя), предназначенный для связи с оператором РР, и формирует контейнеры:

$$C_{i1} = E\left(K_{pi}, P_{i1}\right)$$

и

$$C_{i2} = E(K_{si}, P_{i2}),$$

где P_{i1}, P_{i2} – пароли пользователей для защиты соответствующих контейнеров.

Далее пользователь формирует сетевое имя как $S_i = E\left(C, K_{pi} * A_i\right)$, где * – функция смешивания персональной информации и описания пользователя, C – избранная константа.

Приведем пример формирования данных при помощи алгоритма шифрования «Кузнечик».

Например, при задании A_i =Alisa Valerevna Melnikova при некотором значении ключа получим S_i=b944928487491bde8f5bba9a64b33f4d.

В данном случае сетевое имя имеет длину 32 шестнадцатеричных знака (16 байт), что соответствует длине блока открытого текста алгоритма «Кузнечик».

После формирования сетевого имени и контейнера $C_{i2} = E(K_{si}, P_{i2})$ эти данные синхронизируются между пользователем и оператором РР.

Это означает, что контейнер C_{i2} может быть сформирован и оператором РР и передан пользователю при его регистрации, возможно, выполненной в рамках национального законодательства, при этом пароль для раскрытия контейнера передается лично пользователю при физическом посещении представителя оператора и авторизации пользователя с предъявлением соответствующих документов. Пароль может быть передан и по альтернативным каналам связи, например, СМС при регистрации пользователя с учетом номера его мобильного устройства.

Для подготовки данных для отправки их в РР пользователь может использовать конструктор атомов РР [6], позволяющий создать зашифрованный, подписанный (снабженный имитовставкой) или открытый блок данный. При этом зашифрованный или подписанный блок формируется на персональном ключе пользователя и доступен только самому пользователю, что позволяет обеспечить дополнительно невозможность ознакомления оператора РР с информацией пользователя либо возможность контроля неизменности информации, отправленной оператору РР.

Остановимся подробнее на этом важном свойстве. Наличие двух ключей – K_{pi} – персонального ключа (персональная информация) i-го пользователя РР и K_{si} – сетевого ключа пользователя позволяет достигать свойства эквивалентности симметричного КА электронной подписи. Для этого КА под одной и той же информацией должен быть сформирован дважды, и на K_{si}, тогда оператор может проверить авторство и подлинность, но не сможет изменить содержание, поскольку оно зафиксировано пользователем и может быть проверено только им.

Таким образом, двойное снабжение транзакции или ее части двумя КА позволит придать системе свойства электронной подписи, но при этом не использовать ассиметричные криптографические процедуры и не разворачивать инфраструктуру удостоверяющих центров.

Далее пользователь формирует запрос:

$$Z_{ij} = Im\left(\left[INFO_{ij}, S_i, T_k\right], K_{si}\right)$$

и направляет его на сервер приема-выдачи данных. Сервер приема данных проверяет имитовставку пользователя под запросом, тем самым проводя как аутентификацию отправителя, так и проверку целостности данных.

В том случае, если *INFO* содержит имена отправителя и/или получателя информации S_{si} и S_{rj} соответственно, для этих имен выполняются процедуры:

$$y_1 = E\left(S_{si}, K_{pi}\right)$$

и

$$y_2 = E\left(S_{rj}, K_{pi}\right).$$

Данная процедура делает недоступной для нарушителя имена отправителя и получателя, а оператор РР имеет возможность, используя процедуру расшифрования D и ключ K_{pi}, получить реальные имена пользователей.

Можно предложить интересный метод последовательной генерации сетевых имен, используя итеративную процедуру $y_{10} = E\left(S_{si}, K_{pi}\right)$, $y_{11} = E\left(y_{10}, K_{pi}\right)$ и так далее необходимое число раз при осуществлении каждой новой транзакции. Тогда пользователь и оператор могут восстановить любое имя из цепочки. При этом система получает свойство «нулевого внутреннего разглашения», когда оператор для получения сетевого имени S_{si} должен перебрать ключи K_{pi} для всех известных ему пользователей и пройти назад до S_{si}

по приведенной итеративной цепочке имен y_{1k}, чтобы провести проверку КА. Таким образом, в систему добавляется элемент майнинга, трудоемкость которого зависит от количества пользователей и числа транзакций, совершенного пользователем, например, при количестве пользователей 10^5 и количестве транзакций 10^4 трудоемкость работы оператора РР в системе с нулевым внутренним разглашением составит 10^9 операций, что может существенно снизить быстродействие.

При положительном результате проверки информация передается серверу записи в РР, который передает информацию для обработки в сервер оператора РР, хранящий ключ оператора K_o. Данный сервер записывает в систему хранения данных (СХД) блок Z_{ij} (цифровой конверт). При положительном результате записи в СХД для пользователя формируется квитанция KV_{ij}, содержащая номер блока, куда помещена информация пользователя, номер транзакции, время помещения в РР и подпись оператора под данными пользователя.

Приведем пример такой квитанции.

Dnum:3
Tnum:c09f9ae8a8921d91b41691c061cd6b61
Sign:ad3fed45df10834c
File:a01
NetName:b944928487491bde8f5bba9a64b33f4d
AddTime:01:37:11 13.01.2019

В данном случае квитанция удостоверяет для пользователя с сетевым именем NetName помещение файла a01 в звено распределенного реестра с номером 3, при этом имитовставка в СХД, выработанная оператором РР, принимает значение *Sign*, а номер транзакции составляет значение *Tnum*, время формирования записи (звена РР) AddTime.

Для извлечения данных или для изменения прав доступа к записи (по умолчанию доступ к записи предоставляется пользователю, который ее выполнил) пользователь использует специальные запросы.

Приведем примеры таких запросов.

Access
dnum:1
+:b944928487491bde8f5bba9a64b33f4d

Запрос означает, что доступ к записи с номером 1 дополнительно предоставлен (+) пользователю с приведенным выше сетевым именем «Alisa».

Или
load
dnum:1

Запрос означает, что из РР будет выгружена запись с номером 1.

При запросе на извлечение данных или изменение прав доступа запрос V_m снабжается имитовставкой пользователя и передается на сервер оператора РР, который обращается к СХД в режиме чтения и по номеру записи либо другой информации поиска (сетевому имени, дате) извлекает информацию и передает серверу приема-выдачи данных, либо формирует квитанцию о неуспешном поиске и невозможности извлечения данных.

3. Структура системы хранения данных

В целях обеспечения основных свойств распределенного реестра [7], следующих из названия технологии blockchein – «цепь» или «цепочка» блоков, блокчейн в первую очередь должен обеспечивать свойства цепи – неразрывность и прочность, которые являются парафразом свойства целостности.

Неразрывность определяется как свойство следования блоков (звеньев цепи) одного за другим, в заданной в процессе создания блокчейна последовательности, а прочность – невозможность замены или удаления звена из цепочки.

Если рассматривать блокчейн как системную целостность, то он должен состоять из отдельных элементов – звеньев, каждое из которых в свою очередь делится на элементарные компоненты (назовем их атомами блокчейна). В данном случае для СХД формируется последовательность записей в нотации языка С:

```
l1=fwrite(dnum   ,1,    16,fl);
l2=fwrite(ntran  ,1,    16,fl);
l3=fwrite(tdt    ,1,     8,fl);
l4=fwrite(buf    ,1,buflen,fl);
l6=fwrite(imi    ,1,     8,fl);
l5=fwrite(&buflen,4,     1,fl);
l7=fwrite(dnum1  ,1,    16,fl);
```

dnum – упомянутый выше номер записи (номер звена),

ntran – номер транзакции,

tdt – время и дата формирования звена,

buf – данные пользователя, записываемые в СХД длиной buflen,

$dnum1 = dnum + 1$, обеспечивающая неразрывность перехода к следующему звену,

$imi = Im\langle dnum|ntran|tdt|buf, K_o\rangle$ – имитовставка от конкатенации приведенных выше данных.

Кроме того, старшая область поля dnum заполняется значением *imi* от предыдущего звена, что позволяет добиться того же свойства, как и в блокчейне Bitcoin – зависимости значений хеш-кодов (в данном случае вычисленных при помощи приватного элемента K_o) от всей последовательности предыдущих данных.

Приведем пример последовательного формирования описанных полей:

Dnum:13
Полное значение: 6c006896973848d7000000000000000d
Tnum:44f98f920985679580a8b7bee17de548
Sign:dabfd993c75f3366
File:a01
AddTime:01:10:48 20.01.2019
и
Dnum:14
Полное значение: dabfd993c75f3366000000000000000e
Tnum:ec2120a826f51e32255daa1f6002f5a2
Sign:baa49fb79db3805b
File:a01
AddTime:01:11:09 20.01.2019

Как легко видеть, поле *imi* предыдущего блока заполняет старшие разряды (8 байт) поля dnum текущего блока (предыдущий блок с номерами 13, текущий с номером 14), что позволяет обеспечить зависимость от всей предыдущей информации, помещенной в распределенный реестр.

4. Практическая реализация описанных методик в виде платформы

4.1. Обмен данными между клиентом и сервером

Обмен данными между клиентскими и серверным приложениями осуществляется путём передачи сообщений, имеющих следующий набор блоков данных, располагающихся непосредственно друг за другом в теле передаваемого сообщения:

- заголовок сообщения (32 байта);
- время создания сообщения (8 байт);
- сетевое имя автора сообщения (16 байт);
- размер блока, содержащего текстовую команду (4 байта);
- размер блока, содержащего бинарные данные (4 байта);
- команда в текстовом формате, описывающая действие, которое получатель команды должен осуществить (добавить запись в реестр, извлечь запись из реестра, изменение прав доступа к записи и пр.). Формат и примеры описываются ниже:
- произвольные данные (payload) (это, например, может быть непосредственно содержимое добавляемой в реестр записи);
- подпись для верификации сообщения.

4.2. Формат текстовой команды

Текстовые команды в передаваемых сообщениях состоят из последовательности строк, первая из которых содержит идентификатор команды (ADD, GET, ACCESS и т.д.), а остальные - пары «ключ-значение», для передачи имён параметров их значений.

Примеры команд.

Команда для добавления записи в реестр:

ADD

file: file_name.txt

Команда на извлечение записи из реестра:

GET

dnum: 0000000000000001

4.3. Классы, реализующие работу с примитивами, использующимися при написании кода библиотеки:

- CUserMasterKey - мастер ключ пользователя (размером 32 байта);
- CUserKey - ключ шифрования;
- CKeyContainer - ключевой контейнер;
- CReceipt - квитанция, выдаваемая пользователю при записи данных в реестр;
- CMessage - сообщение для передачи данных и команд между клиентом и сервером.

4.4. Описание и реализация API для создания приложений, использующих библиотеку.

Ниже даётся краткое описание интерфейсов и базовых классов API:

IDB

Данный интерфейс описывает минимально необходимый набор функций, которые должны быть реализованы в создаваемом серверном хранилище, для нормальной работы сервера реестра.

Через данный интерфейс сервер реестра осуществляет:

- получение сетевых ключей и паролей пользователей системы;

- добавление и извлечение записей реестра.

IServerClient

Описание интерфейса, через который сервер реестра осуществляет передачу сообщений клиенту в процессе обработки пришедшего от него сообщения.

Если в процессе обработки пришедшего сообщения серверу необходимо отправить клиенту некоторые данные (например, обрабатывается запрос на извлечение записи из реестра и необходимо отправить клиенту содержимое записи), то делается это через данный интерфейс.

CClient

Реализация базового класса для создания клиентских приложений для взаимодействия с реестром.

Алгоритм создания клиентских приложений на основе данного класса описан ниже.

CServer

Реализация базового класса для создания сервера реестра.

Алгоритм создания серверных приложений на основе данного класса также кратко описан ниже.

Создание приложений

Ниже будет описан процесс создания приложений (клиентской и серверной частей) на основе библиотеки.

4.5. Создание клиентского приложения

Для реализации клиентского приложения необходимо создать класс-наследник от CClient

Авторизация пользователя

Авторизация пользователя на локальной машине происходит через вызов функции login, которой передаются сетевой пароль, сетевое имя и ключевой контейнер пользователя. Таким образом, классы, которые объявлены в приложениях и будут наследоваться от CClient, должны получить эти данные от пользователя (ввод в окне, чтение из файла на локальном компьютере, чтение из токена) и передать их в данную функцию.

Передача запросов на сервер

Для передачи запросов на добавление данных в реестр, реализованы функции:

- send_file - для отправки на сервер содержимого файла с локальной машины;
- send_data - для отправки на сервер произвольных данных, хранящихся в памяти.

Для отправки на сервер запроса на извлечение данных реализована функция get_data, принимающая в качестве параметра строковый идентификатор записи в реестре.

Обработка ответов сервера

Приложение должно самостоятельно реализовывать определение наличия входящих данных от сервера реестра и их получение (постоянное прослушивание сокета или периодический «пинг» сервера на предмет наличия новых сообщений).

После получения данных от сервера реестра приложение должно вызывать функцию parse_message, осуществляющую парсинг и валидацию пришедших данных. В случае удачного парсинга, данной функцией вызывается виртуальная функция on_message, предназначенная для обработки поступившего сообщения.

Реализация функции on_message в классе-наследнике CClient и является реализацией реакции программы на сообщения, поступающие от сервера реестра.

Таким образом, создание клиентского приложения сводится к следующему набору задач:

- получение сетевого имени, пароля, ключа (через пользовательский ввод, чтение из файла, получение через токен) и последующий вызов login для авторизации;
- запрос у пользователя данных для отправки на сервер (окно ввода, выбор файла на локальной машине) и последующий вызов send_data/send_file;
- обеспечение непосредственной передачи данных на сервер посредствам реализафункцию send (отправка данных через tcp, http, файлы в общей папке);
- обеспечение получения данных от сервера (например, посредством прослушивания сокета после отправки сообщения) и последующий вызов parse_message для парсинга и валидации пришедших данных;
- непосредственное описании реакции на пришедшие от сервера данные (отображение пользователю в окне, запись в файл на диск) при реализации функции on_message, которая будет вызываться в случае удачного парсинга и валидации данных.

4.6. Создание серверного приложения

Для создания серверного приложения необходимо создать класс-наследник от класса CServer. При создании необходимо проинициализировать указатель на интерфейс IDB для работы с серверным хранилищем данных (чтобы сервер имел доступ к сетевым ключам пользователей системы, а также непосредственно к базе данных с записями реестра).

Авторизация оператора

Для авторизации оператора реестра используется функция login, который передаются пароль и ключевой контейнер оператора. Таким образом, на создаваемое приложение возлагается функция получения этих данных у пользователя-оператора (непосредственный ввод, чтение из файла или токена).

Получение и обработка запросов

Реализуемое серверное приложение должно самостоятельно обеспечить определение наличия и получение данных запросов пользователей реестра (прослушивание и чтение из сокета или периодическая проверка общей папки на наличие файлов, ожидающих обработку или иной способ получения данных от клиентов).

После получения входящих данных вызывается функция process, которая осуществляет парсинг и валидацию пришедших данных. Для связи с клиентом (в случае появления необходимости отправить ему данные) в процессе обработки запроса данной функции передаётся указатель на объект, реализующий интерфейс IServerClient.

В классе CServer реализована реакция сервера на поступление запросов на добавление и извлечение данных. Если у создаваемого приложения есть необходимость расширить список поддерживаемых команд, в нём необходимо реализовать собственную реализацию виртуальной функции exec_cmd.

Выводы

Сформулированная концепция создания доверенного защищенного распределенного реестра может являться методологической основой для формулирования ведомственных или национальных регулирующих требований в области цифровой экономики [8], а также послужить технической основой для разработки конкретных проектов в области защищенных систем, использующих распределенные реестры в сфере государственного управления, финансов и учетно-сервисных систем.

СПИСОК ЛИТЕРАТУРЫ:

1. Zaitsev A.V., Gostev S.S, Cherkashin P.A., Shcherbakov A.Yu. (2017) Regarding the Technology of Distributed Storage of Confidential Information in Centers of General-Purpose Data Processing, Automatic Documentation and Mathematical Linguistics. Vol. 51, №. 3, P. 117–119. ISSN 0005-1055. DOI: 10.3103/S0005105517030074.
2. Nakamoto, S. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. bitcoin.org. URL: https://bitcoin.org/bitcoin.pdf (дата обращения: 21.05.2019).
3. Биктимиров М.Р., Щербаков А.Ю. Проблемы синтеза доверенных систем // Труды ИСА РАН. 2010. Том 53(4). С. 264–271.
4. Централизованные криптовалюты (2017). URL: geektimes.ru/company/waves/blog/289379/ (accessed: 21.05.2019).
5. Правиков Д.И., Щербаков А.Ю. Изменение парадигмы информационной безопасности // Системы высокой доступности, 2018, № 2, С. 35-39. ISSN 2072-9472. URL: https://elibrary.ru/item.asp?id=35256127 (дата обращения: 21.05.2019).
6. Щербаков А.Ю. О разработке средств для формирования корпоративного распределенного реестра (Блокчейн) // Научно-техническая информация. Серия 2: Информационные процессы и системы, 2018, № 4, С. 30-34. ISSN 0548-0027. URL: https://elibrary.ru/item.asp?id=32850527(дата обращения: 31.05.2019).
7. Биктимиров М.Р., Домашев А.В., Черкашин П.А., Щербаков А.Ю. Блокчейн: универсальная структура и требования // Научно-техническая информация. Сер. 2. 2017. № 11. С. 1–4. URL: https://elibrary.ru/item.asp?id=32597362 (дата обращения: 21.05.2019).
8. Щербаков А.Ю. Синтез универсальной архитектуры и протокола криптовалюты в рамках национального проекта // Системы высокой доступности, № 3, т. 13, 2017. С. 15–18. URL: https://elibrary.ru/item.asp?id=30554617 (дата обращения: 21.05.2019).

REFERENCES:

[1] Zaitsev, A. V., Gostev, S. S., Cherkashin, P. A., Shcherbakov, A.Yu. (2017) Regarding the Technology of Distributed Storage of Confidential Information in Centers of General-Purpose Data Processing, Automatic Documentation and Mathematical Linguistics, Vol. 51, №. 3, 117–119. ISSN 0005-1055. DOI: 10.3103/S0005105517030074. URL: https://bitcoin.org/bitcoin.pdf (accessed: 21.05.2019).
[2] Nakamoto, S. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. bitcoin.org. URL: https://bitcoin.org/bitcoin.pdf (accessed: 21.05.2019).
[3] Biktimirov, M. R., Shcherbakov, A. Yu. Problems of synthesis of trusted systems. Proceedings of ISA RAS. 2010. Vol. 53(4). P. 264–271.
[4] Centralized cryptocurrencies (2017). URL: geektimes.ru/company/waves/blog/289379/ (accessed: 21.05.2019).
[5] Pravikov, D.I., Shcherbakov, A.Yu. (2018) Changing the paradigm of information security, Highly available systems, 2, 35-39. ISSN 2072-9472 URL: https://elibrary.ru/item.asp?id=35256127 (accessed: 21.05.2019).
[6] Shcherbakov, A.Yu. (2018) About development tools for creation corporative distributed ledger (blockchain) Automatic Documentation and Mathematical Linguistics, 2018, №4. P. 30–34. ISSN 0548-0027. URL: https://elibrary.ru/item.asp?id=32850527 (accessed: 21.05.2019).
[7] Biktimirov, M.R., Domashev, A.V., Cherkashin, P.A., Shcherbakov, A.Yu. Blockchain: universal design and the requirements of Scientific and technical information. Ser. 2. 2017. № 11. P. 1–4. URL: https://elibrary.ru/item.asp?id=32597362 (accessed: 21.05.2019).
[8] Shcherbakov, A. Y.. Synthesis of a generic architecture and protocol of cryptocurrencies in the framework of the national project High Availability Systems, №. 3, Vol. 13, 2017. P. 15–18. URL: https://elibrary.ru/item.asp?id=30554617 (accessed: 21.05.2019).

Поступила в редакцию –11 апреля 2019 г. Окончательный вариант – 30 мая 2019 г.
Received – April 11, 2019. The final version – May 30, 2019.