УДК [004.75.056:004.41]:005.92

А.Ю. Щербаков

Комплексный подход к созданию платформы доверенного документооборота с электронной подписью

Рассматриваются возможности создания платформы доверенного документооборота с электронной подписью, предназначенной для взаимной проверки и признания документов, сформированных в различных юрисдикциях и снабженных электронной подписью с использованием различных алгоритмов и ключей. Представлен способ надежного доверенного хранения, обмена и проверки документов на основе доверенного распределенного реестра с использованием смарт-контрактов и симметричных криптографических алгоритмов.

Ключевые слова: *код аутентификации, распределенный реестр, блокчейн, протокол, ключи, электронная подпись, средства криптографической защиты информации, безопасность, датчик случайных чисел, удостоверяющий центр, смарт-контракт*

DOI: 10.36535/0548-0027-2020-11-3

ВВЕДЕНИЕ

Задача взаимной проверки электронных подписей (ЭП)[1], сформированных по различным алгоритмам в различных юрисдикциях, является сегодня весьма актуальной и, к сожалению, пока технически нерешенной.

Предположим, что имеется несколько участников системы, находящихся в различных государствах (юрисдикциях) и имеющих различные алгоритмы и ключи электронной подписи, для которых необходима общая возможность подтверждения документов, подписанных этими подписями [1]. Классическое решение этого вопроса, связанное с созданием единого доверенного удостоверяющего центра, практически невозможно, поскольку затруднительно выработать общие технические и юридические регламенты работы центра. Кроме того, в силу закрытости процедур оценки качества криптографических механизмов, участники системы находятся в весьма затруднительном положении при опубликовании результатов криптографических исследований качества электронных подписей.

При этом мы понимаем, что оценке надежности подвергаются как совокупность алгоритмов ЭП, так и свойства удостоверяющего центра, который формирует сертификаты – подписывает своей электронной подписью открытые ключи пользователей (участников системы). Кроме того, сами процедуры по проверке ЭП в системе часто проводятся автоматизировано – с участием смарт-контрактов.

Известно решение, называемое «служба доверенной третьей стороны» (служба ДТС – *Litoria DVCS*) [2], которое, по мнению его авторов, обеспечивает юридическую значимость квалифицированной электронной подписи на иностранном алгоритме в юрисдикции Российской Федерации, проверяя эту иностранную подпись и выдавая юридически значимую квитанцию с результатами проверки. Юридическая значимость квитанции обеспечивается электронной подписью, выполненной с помощью отечественного криптографического алгоритма.

1. Иностранная электронная подпись передается на проверку в службу доверенной третьей стороны.

2. ДТС определяет криптографические алгоритмы, на которых выполнена ЭП и передает эту электронную подпись в ДТС национального сегмента, который может легитимно работать с данными криптографическими алгоритмами.

3. ДТС национального сегмента проверяет электронную подпись, **создает квитанцию с результатами проверки** и передает квитанцию в ДТС сегмента Российской Федерации (ДТС РФ).

4. ДТС РФ подписывает квитанцию отечественной электронной подписью, тем самым придавая юридическую значимость иностранной ЭП, результаты проверки которой зафиксированы в квитанции.

5. ДТС РФ возвращает подписанную квитанцию пользователю.

Очевидно, что предлагаемая схема имеет слабое место в п.3, когда квитанция передается в ДТС РФ,

[1] Федеральный закон "Об электронной подписи" от 06.04.2011 N 63-ФЗ. – URL: http://www.consultant.ru/document/ cons_doc_LAW_112701/

а проверить электронную подпись под ней не представляется возможным.

Далее мы покажем, что применение распределенного реестра в совокупности с доверенным смарт-контрактом позволит решить эту проблему.

Электронная подпись включает три криптографических алгоритма: 1) **алгоритм хеширования**, который преобразует подписываемый текст (документ) в вектор фиксированной длины (результат хеширования); 2) **алгоритм подписания (простановки ЭП)**, оперирующий с результатом хеширования, секретным (приватным) ключом пользователя и некоторым случайным числом; 3) **алгоритм проверки ЭП**, оперирующий с результатом хеширования и открытым ключом (сертификатом) пользователя.

РЕШЕНИЕ ЗАДАЧИ ВЗАИМНОЙ ПРОВЕРКИ ЭЛЕКТРОННЫХ ПОДПИСЕЙ

Предлагается следующее решение задачи построения платформы документооборота с возможностью проверки различных электронных подписей на базе доверенного распределенного реестра (ДРР), который реализует платформу доверенного документооборота. Доверенность понимается в смысле сохранения свойств документов в системе на протяжении их жизненного цикла. При этом платформа имеет и архивную функцию, поскольку удаление звеньев распределенного реестр (РР) невозможно.

Каждый участник документооборота регистрируется в качестве участника в системе РР и получает возможность размещать и получать документы в этом реестре. При этом целостность и авторство документов проверяется оператором РР при помещении документов в реестр. В распределенном реестре документы помещаются с локальной электронной подписью своего владельца.

Таким образом, в распределенном реестре каждый документ снабжен как подписью владельца, так и кодом аутентификации (КА) РР, который позволяет убедиться в целостности информации и установить ее принадлежность. В качестве кода аутентификации предлагается функция $I=Im(x, k)$ – функция вычисления имитовставки от информации x на ключе k. Эта функция обладает возможностью как авторизации пользователя, так и контроля целостности передаваемой и хранимой информации. Она построена на симметричном криптографическом алгоритме и при этом не является средством криптографической защиты информации, на которое распространяются экспортные и импортные ограничения.

Кроме того, пользователи могут открывать доступ к своим документам другим участникам, а оператор РР может направить документ для проверки электронной подписи тому, кто может выполнить эту операцию, исходя из наличия ключей проверки (сертификатов) ЭП и соответствующих криптографических алгоритмов, и получить квитанцию о верности или неверности ЭП, которая также защищена кодом аутентификации и помещена в распределенный реестр.

Таким образом формируется общая инфраструктура доверия, в которой каждый документ подписан как минимум одной ЭП и кодом аутентификации участника, что позволяет убедиться в подлинности информации и анализировать квитанции о верности ЭП, также защищенные КА. Дополнительно все документы образуют единую цепочку с невозможностью исключения документа из нее, имеют метки времени и сквозную нумерацию. Вся система образует доверенную СУБД с возможностью разветвленного поиска и предоставления различных выборок и статистик в соответствии с правами и ролями участников.

Регламент работы системы, включающий помещение документов в распределенный реестр и запросы на проверку электронной подписи, может быть описан достаточно простым гражданско-правовым договором, который в идентичной форме заключается в каждой юрисдикции и описывает не использование ЭП, а движение документов в системе распределенного реестра.

Введем дополнительные термины для описания дальнейших способов решения задачи создания платформы.

Смарт-контракт (СК) – последовательность команд и вызовов исполняемых модулей из хранилища приложений.

Компилятор смарт-контракта – программный модуль, который переводит команды и имена модулей в сжатую кодированную форму (байт-код).

Хранилище приложений – область данных сервера СК, в которой находятся приложения для выполнения с предварительной проверкой их имитовставки (кода аутентификации) на ключе вызывающего пользователя.

Предположим, что смарт-контракт представляет собой последовательность команд и вызовов исполняемых модулей из хранилища приложений, выполняется на отдельном сервере (сервер смарт-контрактов) и связан по данным с сервером оператора распределенного реестра (сервер оператора РР).

Смарт-контракт формируется пользователем и направляется в систему как стандартный элемент распределенного реестра, описанный выше обычным порядком.

В системе вводятся три вида смарт-контрактов: текстовый, байт-код и анонимизированный байт-код, в последнем вызовы приложений анонимизированы и не видны другим пользователям.

Сервер оператора выделяет смарт-контракт из входного потока данных и направляет его на сервер СК, где он верифицируется и, при положительном результате верификации, исполняется.

Верификация заключается в проверке наличия вызываемых приложений в хранилище приложений и их доступности для данного пользователя, а также в правильном синтаксисе написании команд.

При возникновении ошибки в некоторой строке смарт-контракта сервер СК формирует квитанцию, где указывает код ошибки и строку, в которой она произошла, и направляет эту квитанцию на сервер оператора РР.

После успешного выполнения смарт-контракта также формируется квитанция для пользователя, направившего смарт-контракт в систему.

Приложения, доступные для пользователя, в обязательном порядке подписаны имитовставкой (кодом аутентификации) на его ключе.

КРАТКОЕ ОПИСАНИЕ ТЕХНОЛОГИИ ПЛАТФОРМЫ

Пользователь является абонентом системы РР, оператор которого располагает **модулем доверенного хранения пользовательских ключей** (МДХПК), а подписанные электронной подписью и кодом аутентификации документы хранятся в распределенном реестре [3].

МДХПК при помощи качественного датчика случайных чисел формирует ключи КА, хранит их без выдачи во внешнюю среду (неизвлекаемо) и выполняет на них функции проверки кода аутентификации.

Аутентификация пользователя производится с применением присылаемых на мобильное устройство одноразовых паролей, которые вводятся в программы связи с сервером и используются для реализации протоколов аутентификации. При необходимости на мобильное устройство может быть передан ключ для создания защищенного сеанса с сервером, закрытый на одноразовом пароле.

Понятие «неизвлекаемость» на современном техническом уровне можно трактовать так: в хранилище, понимаемом как изолированное техническое устройство, нет возможности прочитать загруженный или сформированный ключ – по причине отсутствия программных и технических интерфейсов извлечения ключа во «внешний мир». Также невозможно извлечение любой информации о ключе, существенно раскрывающей его содержание, и отсутствует (в техническом плане) информация о ключе в технических каналах наблюдения (электромагнитном, акустическом, визуальном).

Кроме того, модуль доверенного хранения пользовательских ключ должен обеспечивать выполнение криптографических операций (преобразований) на загруженном в него ключе, а также выработку собственных ключей без их извлечения и выдачу вовне только результатов криптографических операций.

При соблюдении свойств «необратимой или сингулярной» загрузки ключей (по аналогии с физической «черной дырой», куда информация и материя попадают безвозвратно) процессы управления ключами осуществляются по их номеру или идентификатору. При этом для хранения идентификаторов возможно применение распределенного реестра, а для верификации данных участников системы, их аутентификации и разрешения споров – использование ключей.

Важнейшим моментом этой технологии являются механизмы качественной генерации ключей, т. е. корректной работы датчиков случайных чисел и проверки качества случайных последовательностей.

Термины и обозначения для формального описания алгоритма платформы

Xi – пользователь корпоративной системы.

NMOi – номер мобильного устройства пользователя.

Ai –информация, описывающая пользователя Xi.

Kxi – персональный ключ пользователя, неизвестный самому пользователю и хранимый в МДХПК.

Ksi – сетевой ключ пользователя (также являющийся частью персональной информации пользователя), предназначенный для связи с оператором РР.

Ci – ключевой контейнер пользователя, представляющий собой персональную информацию пользователя (сетевой ключ), закрытый на пароле пользователя при помощи обратимой криптографической процедуры.

Si – сетевое имя пользователя, однозначно связанное с Ai.

INFOij – информация *i*-го пользователя, сформированная на его рабочем месте и направляемая для хранения и обработки, имеющая условный номер *j* и содержащая ЭП, зависящую от этой информации.

Kvij – квитанция, сообщающая о результате обработки *j*-го информационного блока для *i*-го пользователя.

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

Напомним, что функция вычисления имитовставки обладает возможностью как авторизации пользователя, так и контроля целостности передаваемой и хранимой информации.

Краткое описание команд смарт-контракта

Команда загрузки данных из РР

rload номер_реестра номер_записи файл

где:

номер_реестра – десятичный номер реестра, из которого извлекаются данные,

номер_записи – десятичный номер извлекаемой записи,

файл – имя файла, куда извлекается запись.

Команда возвращает код ошибки или 0 при успешном выполнении.

Команда выгрузки данных в РР

upload файл номер_реестра имя_пользователя номер

где:

файл – имя файла, загружаемого в реестр,

номер_реестра – десятичный номер РР,

имя_пользователя – имя пользователя, которому предназначена формируемая запись,

номер – имя файла, в который при успешном завершении помещается номер сформированной записи (звена РР).

Команда возвращает код ошибки или 0 при успешном выполнении.

Общие команды пакетных файлов

Copy, del, md, cd, ren, dir также поддерживаются в смарт-контракте.

Команда вызова приложений

имя_приложения аргументы,

где:

имя_приложения – имя вызываемого приложения из хранилища приложений,

аргументы – названия аргументов или файлов, с которыми работает приложение.

Команда создания квитанции

mkk файл имя_пользователя,

где:

файл – файл содержащий тело квитанции,

имя_пользователя – имя пользователя, которому предназначена квитанция.

Например,

```
rem Перевод с кошелька k1 пользователя
7d5402af на кошелек пользователя 20ef84c5
rload 1 8146 k1
rload 1 9378 k2
perevod 50 k1 k2
upload k1 1 7d5402af num1
upload k2 1 20ef84c5 num2
```

В приведенном примере смарт-контракта использовано приложение-перевод (perevod), которое переводит 50 условных единиц с кошелька k1, загруженного из записи 8146 реестра 1, на кошелек k2, загруженного из записи 9378 также реестра 1, т. е. k2 = k2 + 50, k1 = k1 - 50, после чего записи в реестре обновляются, номера новых записей помещаются в файлы num1 и num2, соответственно. Квитанции в данном случае могут быть сформированы автоматически оператором РР, либо сервером СК.

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

Введем следующие обозначения:

y=E(x, k) – алгоритм зашифрования данных x на ключе k,

x=D(y,k) – алгоритм расшифрования данных x на ключе k,

Pi – пароль пользователей для защиты соответствующих контейнеров.

Пользователь формирует запрос Zij = Im([Si,DATAj], Ksi) и направляет его в распределенный реестр. Сервер проверяет имитовставку пользователя по запросу, тем самым проводя как аутентификацию отправителя, так и проверку целостности данных DATAj, в состав которых могут входить и подписанные электронной подписью данные, описанные нами далее.

При положительном результате проверки запроса информация передается в распределенный реестр и модуль доверенного хранения пользовательских ключей. При положительном результате записи обработанных данных в средствах хранения данных для пользователя формируется квитанция Kvij, содержащая номер блока, куда помещена информация пользователя, номер транзакции, время помещения в распределенный реест и имитовставка на Ksi под указанными данными квитанции пользователя.

Описание протокола

Пользователь владеет мобильным устройством, имеющим связь с оператором РР, к которому подключен модуль доверенного хранения пользовательских ключей, формирующий при помощи качественного датчика случайных чисел ключи, хранящий их без выдачи во внешнюю среду и выполняющий на них функции вычисления и проверки кода аутентификации.

Регистрация пользователя

Пользователь при помощи веб-интерфейса – приложения, установленного на его мобильном устройстве, либо посылкой СМС направляет запрос на регистрацию в системе. Для этого он указывает (для СМС эта информация определяется автоматически) номер своего мобильного устройства NMOi. Оператор РР помещает номер пользователя в базу данных, дает команду МДХПК на выработку случайного ключа Ksi для связи пользователя Xi с оператором РР.

Модуль доверенного хранения пользовательских ключей вырабатывает Ksi, проверяет его статистические свойства и при положительном результате проверки запоминает Ksi, экспортирует его в виде контейнера Csi, закрытого на пароле Pi. Затем МДХПК вырабатывает Kxi, проверяет его статистические свойства и при положительном результате проверки запоминает Kxi, а также вырабатывает сетевое имя Si, определяющее идентификаторы ключей Ksi и Kxi, используемое для операций с пользовательскими данными пользователя Xi с именем Si.

Пароль Pi передается пользователю на материальном носителе (карта памяти, помещаемая в мобильное устройство) или присылается по СМС. Ci также может быть записан на карту памяти, либо выслан по открытым каналам, либо помещен во внешнюю (облачную) систему хранения данных.

Пользователь имеет возможность изменить Pi (с изменением контейнера Ci).

Обмен информацией

Пользователь Xi передает информацию INFOk пользователю Xj.

Для этого он зашифровывает INFOik на ключе Ksi (извлеченном из Ci) и передает E(Ksi, INFOik) в РР, помещающий принятую информацию в МДХКП, который, в свою очередь, расшифровывает информацию внутри себя, зашифровывает на Kxi и передает E(Kxi, INFOik) в корпоративную или внешнюю систему хранения (возможно, в корпоративный распределенный реестр или в облако). При помещении E(Kxi, INFOik) в корпоративную систему хранения формируется квитанция, содержащая номер (ссылку) #INFOik в системе хранения, которая доставляется отправителю Xi, а #INFOik сообщается получателю Xj.

Ключ Kxi никому не известен и всегда находится внутри модуля доверенного хранения пользовательских ключей.

При запросе пользователя Xj на прочтение информации E(Kxi,INFOik) выполняется расшифрование E(Kxi,INFOik) и зашифрование INFOik на ключе Ksj внутри МДХКП, после чего результат передается пользователю Xj, который расшифровывает его на своем мобильном устройстве при помощи Ksj.

При хранении всех трех единиц – E(Kxi,INFOik), E(Ksi,INFOik) и E(Ksj,INFOik) –система получает полноценные свойства электронной подписи, с возможностью проверки в МДХПК и выдачи обоим пользователям результата верификации данных.

Если же участники системы не имеют возможности использовать модуль доверенного хранения пользовательских ключей, либо шифрование данных нецелесообразно, либо использование алгоритмов шифрования связано с экспортными или иными ограничениями, то ранее зарегистрировавшийся в системе участник дает запрос на проверку подписанного электронной подписью массива INFO тому участнику системы, который имеет открытый ключ или сер-

тификат для проверки. Этот участник проверяет ЭП и сообщает результат в виде отдельного массива, который мы назовем *CHECK* (содержащего в обязательном порядке и проверяемый документ) в распределенный реестр, оператор которого при положительном результате проверки ЭП связывает для других участников запись *CHECK* с первичным запросом INFO и тем самым дает возможность другим участникам системы проверить документ сравнивая содержания INFO и *CHECK*.

СМАРТ-КОНТРАКТ И ОРАКУЛЫ

Другим вариантом взаимодействия в системе документооборота может служить следующая схема.

Стороны документооборота договариваются и составляют смарт-контракт, в котором прописывают необходимые условия (в том числе порядок выбора посредника в случае необходимости). Например, что в определенный промежуток времени должен поступить перевод с некоторого адреса на соответствующий аккаунт смарт-контракта на определенную сумму и что в определенный промежуток времени в учетной системе РР должны появиться соответствующие блоки.

Если все условия выполнены, то средства будут переведены, если нет, то в течение определенного периода времени они возвращаются на исходный адрес. При таком механизме расчета минимизируются риски мошенничества любой из сторон и исключается необходимость доверия.

Важно отметить, что смарт-контракт может оперировать только теми данными, создание, хранение и обработка которых осуществляется в пределах одной учетной системы РР. Это означает, что валидаторы должны иметь доступ к данным, которые используются в смарт-контракте (такие как токены, балансы, транзакции, временные метки и др.). Если средства, участвующие в переводе, не учитываются в блокчейне, то необходимо использовать оракулы, т. е. такие доверенные третьи стороны, которые получают информацию от *offchain*-ресурсов и поставляют ее в блокчейн. Здесь возникает вопрос о достоверности и полноте этих данных. В настоящее время самыми распространенными подходами к решению этой проблемы являются консенсус оракулов (примером может служить проект Chainlink) или использование TLSNotary-доказательств для подтверждения корректной работы оракула. Необходимо подчеркнуть, что оракул является именно поставщиком данных, а не их источником. Поэтому, несмотря на то, что оба подхода в некоторой степени гарантируют корректность передачи данных от источника к смарт-контракту, они не могут обеспечить достоверность данных самого источника. Как следствие, рекомендуется обращаться к нескольким доверенным источникам данных.

Использование оракулов требует доверия пользователей к передаваемым данным. Эти данные должны быть подписаны электронной подписью оракула, а пользователи должны иметь возможность убедиться – какие данные и какой оракул передал на вход смарт-контракту.

Таким образом, с использованием оракулов также нельзя полностью исключить влияния третьей стороны. Оракул является потенциальной точкой отказа, так как существуют риски предоставления ошибочных данных при сбое, злонамеренном действии, компрометации или отказе в обслуживании. Для минимизации уровня доверия рекомендуется использовать нескольких оракулов, а также предусмотреть возможность выбора и обновления списка оракулов и их мотивацию при разработке смарт-контракта. Имеет смысл продумать возможность обновления смарт-контракта (в случае некорректной работы) еще на этапе его проектирования, если такая функциональность не реализована на платформе смарт-контрактов (как например в EOS).

ИСПОЛЬЗОВАНИЕ ДОКАЗАТЕЛЬСТВ С НУЛЕВЫМ РАЗГЛАШЕНИЕМ

Отметим, что в общем случае достаточно сложно обеспечить конфиденциальность входных данных и используемых в смарт-контракте условий.

Одним из наиболее перспективных направлений для обеспечения приватности пользователей считаются доказательства с нулевым разглашением (*zk-proofs*). Существуют интерактивные доказательства, состоящие из последовательности действий запросов и ответов между доказывающим и проверяющим (*commitment-challenge-response*), и неинтерактивные (*Random oracle model*, *Fiat-Shamir heuristic*, *Pairing-based* доказательства и др.), когда от проверяющего не требуется непосредственного опроса доказывающего. Особый интерес для децентрализованных учетных систем представляют именно неинтерактивные доказательства ввиду отсутствия необходимости дополнительного взаимодействия между участниками.

Необходимо заметить, что в зависимости от модели учета внутри системы (UTXO или аккаунты/балансы) для обеспечения валидности транзакций могут быть разные правила. Таким образом, требования и архитектура доказательств с нулевым разглашением систем для реализации транзакций с сохранением конфиденциальности отличаются для каждой модели [4].

В контексте нашей задачи можно использовать zk-SNARKs – неинтерактивные доказательства с нулевым разглашением (Non-Interactive Zero Knowledge protocol, NIZKP), имеющие некоторые ограничения:

- требования к вычислительным ресурсам и размер самих доказательств зависят от конкретного протокола, реализации и математической модели, на основе которых работают данные доказательства. Обычно транзакции, использующие доказательства с нулевым разглашением, имеют гораздо больший размер, что накладывает определенные требования на устройства конечных пользователей;
- zk-SNARKs являются компактными доказательствами, и к их основному недостатку можно отнести требование к наличию процедуры доверенной установки (*trusted setup*) для начальной настройки (при взаимодействии нескольких участников). На этом этапе происходит генерация открытых общесистемных параметров, использование которых позволяет производить проверку транзакций на соответствие правилам протокола. В данном случае верификаторы генерируют специальный секрет, ко-

торый должен быть сразу же уничтожен после доверенной установки, так как его существование допускает публикацию ложных доказательств (*fake proofs*). Однако в реальности не существует способа проверить факт удаления секрета. Эти риски можно снизить, выполняя установку секрета с использованием протокола конфиденциального вычисления (*multiparty computation – MPC*). В этом случае достаточно, чтобы хотя бы один верификатор был честным и удалил свой *toxic waste* после процедуры.

Из-за этого ограничения SNARK-протоколы плохо подходят для произвольных Тьюринг-полных смарт-контрактов, так как каждый новый контракт потребует новой установки. Интересным предложением по имплементации приватных смарт-контрактов может служить система Hawk [5], однако она не только требует новой установки на каждый контракт, но и необходимость использования так называемого «доверенного менеджера», который имеет доступ к приватным данным пользователя. Другой альтернативой реализации приватных смарт-контрактов могут служить СК типа Bulletproofs, так как здесь не требуется проведения процедуры доверенной установки и есть возможность реализовать обобщенный ZKP-протокол с относительно небольшими по размеру доказательствами [6, 7]. Существуют и модификации zk-SNARK [8], не требующие процедуры доверенной установки.

ЗАКЛЮЧЕНИЕ

Предлагаемая в настоящей статье технология создания платформы доверенного документооборота с электронной подписью может стать основой для создания международной системы доверенного документооборота с электронной подписью, предназначенной для взаимной проверки и признания документов, сформированных в различных юрисдикциях и снабженных электронной подписью, а соответственно, и для выполнения юридически значимых действий в рамках обмена электронными документами [9]. За техническую основу платформы может быть взят способ надежного доверенного хранения, обмена и проверки документов на основе доверенного распределенного реестра, с использованием смарт-контрактов и симметричных криптографических алгоритмов.

СПИСОК ЛИТЕРАТУРЫ

1. Электронная подпись: трансграничное взаимодействие. – URL: https://habr.com/ru/post/144401/
2. Трансграничный электронный документооборот. – URL: https://ca.gisca.ru/solutions/transgranichnyy-elektronnyy-dokumentooborot/
3. Кузьменко В.В., Макаров В.Л., Разгуляев К.А., Хан Д.В., Щербаков А.Ю. Новый подход к обеспечению безопасности периметра бизнес-процессов и аутентификации пользователей в корпоративной системе // Вестник современных цифровых технологий. – 2020. – №3. – С.10-13.
4. ZKProof. ZKProof Community Reference. Version 0.2 / eds. by D. Benarroch, L.T.A.N. Brandão, E. Tromer. Pub. by zkproof.org. Dec. 2019. – URL: https://zkproof.org
5. Kosba Ahmed, Miller Andrew, Shi Elaine, Wen Zikai. Charalampos Papamanthou. Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. – URL: https://eprint.iacr.org/2015/675.pdf (дата обращения: 28.04.2020).
6. Bünz B., Agrawal S., Zamani M., Boneh D. Zether: Towards privacy in a smart contract world. – URL: https://crypto.stanford.edu/~buenz/papers/zether.pdf (дата обращения: 23.04.2020).
7. Lin Dr. Suterusu yellowpaper. – URL: https://github.com/suterusu-team/Suter_yellowpaper/blob/master/Suterusu%20yellowpaper%20V%200.2.pdf (дата обращения: 25.04.2020).
8. Bünz Benedikt, Fisch Ben, Szepieniec Alan. Transparent SNARKs from DARK Compilers. – URL: https://eprint.iacr.org/2019/1229.pdf (дата обращения: 20.04.2020).
9. Шаститко А.Е., Шпакова А.А. Регулирование трансграничного электронного документооборота в евразийском экономическом союзе // Государственное управление. Электронный вестник. – 2018 г. – URL: https://cyberleninka.ru/article/n/shastitko-a-e-shpakova-a-a-regulirovanie-transgranichnogo-elektronnogo-dokumentooborota-v-evraziyskom-ekonomicheskom-soyuze/viewer.

Материал поступил в редакцию 16.10.20.

Сведения об авторе

ЩЕРБАКОВ Андрей Юрьевич – доктор технических наук, профессор, руководитель Центра развития криптовалют и цифровых финансовых активов ВИНИТИ РАН, профессор кафедр «Интеллектуальные системы информационной безопасности» технического университета МИРЭА и «Безопасность цифровой экономики и управления рисками» Российского государственного университета нефти и газа имени И.М. Губкина, консультант по криптографическим технологиям фирмы «Лаборатория Касперского», Москва
e-mail: x509@ras.ru