Информационные технологии в системном анализе

Методология хранения электронных документов в децентрализованных блокчейн приложениях (DApps)

О. Хаммуд[I], И.А. Тарханов[II,III]

[I] Национальный исследовательский технологический университет (НИТУ) «МИСиС», г. Москва, Россия
[II] Федеральное государственное бюджетное учреждение высшего образования «Государственный академический университет гуманитарных наук», г. Москва, Россия
[III] Федеральное государственное учреждение «Федеральный исследовательский центр «Информатика и управление» Российской академии наук», г. Москва, Россия

Аннотация. В данной работе исследуется проблема распределенного хранения метаданных электронных документов в децентрализованных системах на базе блокчейн (DApps). Рассматриваются известные способы хранения файлов в распределенных системах, включая централизованное хранилище, торренты, IPFS и Storj. Предлагается подход построения таких систем, который позволяет оптимизировать размер необходимого для хранения места и одновременно обеспечить безопасное хранение данных в географически распределенных системах компании, либо внутри консорциума компаний. Делается расчет надежности данного подхода, а результат сравнивается с традиционным решением на базе полного резервного копирования.

Ключевые слова: *блокчейн, p2p, распределенное файловое хранилище, электронный документ, надежность.*

DOI: 10.14357/20790279210205

Введение

В современном мире крупным компаниям или консорциумам необходимо обмениваться юридически значимыми документами. Поэтому появляется множество систем внешнего электронного документооборота (ВЭД) как в России (Таском, Тензор и т.д.) [1], так и в Европе (Peppol, EDF+) [2]. Крупные корпорации разрабатывают свои собственные системы для распределенного и безопасного обмена данными.

Такого класса системы, созданные на базе централизованной архитектуры (одна база данных), подвержены взлому и неотслеживаемым модификациям [3], но популярны, так как это относительно быстро развертываемое решение, эффективное по стоимости и скорости внедрения. Децентрализованные системы сложно развертываемые и дорогостоящие, но в целом они более надежные несмотря на то, что отдельные ее узлы по-прежнему остаются уязвимыми. Одной из перспективных технологий повышения надежности информационных систем является блокчейн DLT [3]. Приложения, которые взаимодействуют друг с другом не через общую централизованную базу данных, а используют для хранения данных блокчейн называются DApps [4]. Они привлекают все

больше внимания со стороны разработчиков корпоративных решений. Как было сказано выше, главным препятствием для внедрения DApps является высокая стоимость их внедрения, что существенно сокращает круг потенциальных пользователей [5].

Для того чтобы снизить стоимость системы, где данные предполагается хранить в DLT, ее можно объединить с P2P сетью. При этом файлы будут храниться в P2P сети (или IPFS), а метаданные этих файлов – в DLT. Многие исследователи считают такой подход перспективным [6,11]. Если удастся решить проблему более эффективного хранения данных, т.е. уменьшить стоимость внедрения DApps, то это может послужить драйвером для внедрения таких систем в компаниях. Именно разработке метода эффективного хранения данных в DApps посвящено данное исследование.

1. Хранение электронных документов в блокчейн

1.1. Концептуальная архитектура

Рассмотрим задачу создания надежной платформы распределенного хранения юридически значимых электронных документов. Эта платформа должна быть эффективной с точки зрения скорости доступа (в первую очередь, чтения) и использования ресурсов (дискового пространства). Как отмечалось выше, перспективным является подход объединения двух технологий – DLT и распределенного файлового хранилища (например, P2P, torrent и т.д.).

Рассмотрим из чего состоит электронный документ [7]:

- Метаданные (как правило, это реквизиты и служебная информация в XML формате).
- Файлы, связанные с документом (в текстовых или бинарных форматах – DOCX, PDF и т.д.).
- Связь с другими документами (указывается тип связи и ссылка или идентификатор на другой документ).
- Электронная подпись – результат подписания закрытым ключом (как правило, содержит результат шифрования закрытым ключом в виде хеша и открытый ключ, который нужен для расшифровки).

Таким образом, все метаданные документа, связи между документами, метаданные связанных файлов и составные части электронной подписи могут храниться в блокчейн. Непосредственно файлы хранятся в распределенном файловом хранилище (рис. 1).

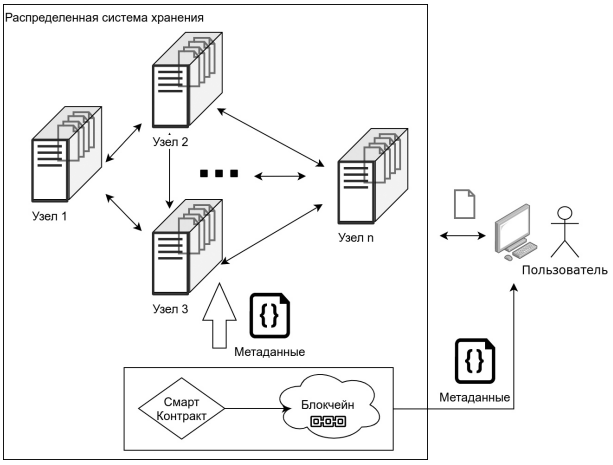

Рис. 1. Концептуальная архитектура системы обмена электронными документами

При запросе пользователем файла, происходит проверка данных файла на целостность и наличие прав у пользователя. При загрузке новой версии файла заполняются его новые метаданные и добавляется запись в блокчейн об этом файле.

1.2. Разделение файлов на блоки

В рассматриваемой системе могут храниться различные типы файлов, включая двоичные, в том числе достаточно большого размера. Для решения задачи оптимизации места хранения каждый файл следует разделить пополам. Тогда его можно было хранить на двух серверах и применить известный подход Стирающего кода (Erasure coding) [8]. В случае хранения больших файлов, они могут быть разделены на блоки для увеличения скорости синхронизации. Однако разделение файлов на боль-

Табл. 1

Files – Структура таблицы хранения файлов в системе

Index	File name	Hash	Timestamp	Creator
/var/data/ myfile1	blockchain.pdf	29ce8a685d966da960645068b0 b36c99eb2ae3c2	1613824311	0xacF5dc45172202D2Ed9721AA4803 0A9d0FB0f0b8
/var/data/ myfile2	123.docx	46303c376da30655cd481db2f7 accdc0e7b617d6	1613029375	0xacF5dc45172202D2Ed9721AA4803 0A9d0FB0f0b8

шие блоки ограничивает операции ввода-вывода. В случае сетевых ошибок требуется возобновить загрузку с последнего блока.

Таким образом, нам необходимо предусмотреть возможность индексации файлов (например, используя для этого сетевые абсолютные пути) и хранить информацию о файле. Для этого предлагается следующая структура размещения файлов в DLT, чтобы можно было в последствии оперировать ей через смарт-контракт. На каждом узле (Node) хранится информация о файлах, которые он хранит, хеш файла во всей системе и путь до него на данном сервере (табл. 1).

Существует также общий массив в DLT с информацией о блоках внутри каждого кластера.

Табл. 2

Parts. Расположение частей файла

Index	Node	Part
/var/data/myfile1	0	1
/var/data/myfile1	1	2
/var/data/myfile1	2	0
/var/data/myfile2	2	1

В табл. 2 – Node это идентификатор узла, а Part указывает, какая часть файла хранится в узле. В примере из табл. 2 у myfile1 части (1 и 2). Part = 0 означает, что в данном месте хранится его стирающий код.

Стирающий код имеет ряд реализаций [8]. Здесь мы рассматриваем самую простейшую форму, которая работает путем вычисления XOR двух файлов. Допустим, у нас есть два файла file1 и file2 и они имеют одинаковую длину L в битах. XOR обоих файлов — это третий файл (file3), который имеет такую же длину (L). Теперь предположим, что каждый из этих файлов находится на разных серверах.

Если сервер, содержащий файл file1, вышел из строя, то его можно вычислить из файлов file2 и file3, используя таблицу XOR. Таким образом, вместо сохранения полного дубликата обоих файлов (что приведет к общей длине 4L) сохранится только один файл вместо двух дубликатов, поэтому общая длина составит 3L. Это означает, что мы можем сэкономить 25% от суммарного объема необходимого места на всех серверах нашей системы.

1.3. Управление версиями файлов

Логично предположить, что используемое для хранения электронных документов файловое хранилище должно иметь поддержку версионности файлов. Обновление файла означает добавление новой его версии, а не замену содержимого существующего файла. При удалении / обновлении файла его запись не удаляется, а переносится в другой массив (OldFiles). Этот массив имеет ту же структуру, что и Files, но дополнительно хранит дату вставки в таблицу.

Поскольку для разных версий двоичных файлов может потребоваться большой объем хранилища, старые версии могут удаляются при наступлении двух правил:

- версия старше определенного времени, определенного в сети блокчейн (например, год);
- это не последняя версия файла.

При сочетании этих двух условий пространство для хранения не тратится на ненужные файлы, и, если авторизованный пользователь по какой-то причине решил уничтожить файл, добавив несколько новых версий, он не будет удален окончательно.

Для каждой версии файла справедливы те же правила хранения, которые описаны в Разделе 1.2.

1.4. Удаление файлов

Операция удаления файла должна быть полной. Должен быть надежный способ, чтобы, если кто-то взломал пользователя, имеющего право редактирования, не смог бы удалить файлы из системы. Этой цели можно достичь, добавив новый массив для файлов, которые необходимо удалить (табл. 3).

Табл. 3

Таблица удаляемых файлов

File ID	Timestamp	User_id	Decision
47	1613029375	13	1
47	1613029375	28	1
391	1203393411	14	1

Таким образом, вместо удаления файла его идентификатор попадает в таблицу, и записывается временная метка удаления.

Кроме того, все пользователи в системе, у которых уже есть доступ к этому файлу, получают уведомление об этом (возможно, с помощью графического интерфейса). Таким образом, пользователи, которые не одобряют эту операцию, могут проголосовать против ее удаления.

1.5. Смарт-контракт управления файлами блоками

Важным элементом предложенной методологии является реализация алгоритма управления разделения файлов на блоки и учет хранящихся блоков. Блокчейн с поддержкой смарт-контрактов [3] можно использовать для определения того, в каком кластере будут находится необходимые данные:

1. При добавлении или обновлении файла клиентское приложение вызывает метод смарт-контракта, который опрашивает существующие кластеры, выясняет, где больше всего свободного места или какой из них имеет самую высокую скорость подключения к текущему приложению (алгоритм выбора может быть параметризован на стороне сервера в кластере).
2. Далее на стороне клиентского приложения происходит вычисление стирающего кода и разделение файла на блоки.
3. Далее на 1-й сервер кластера происходит запись 1-го блока по заранее определенному пути, образованному по file id. Например, /files/var/data/articles/248.pdf/1
4. 2-й сервер аналогично сохранят 2-ую часть файла. Например, /files/var/data/articles/248.pdf/2.
5. То же самое происходит и при добавлении стирающего кода с частью 0 на 3-ий сервер.
6. При обновлении предыдущие записи переносятся в аналогичную таблицу старых версий – oldFiles.
   6.1. Происходит проверка условий для массива oldFiles из пункта 1.3. Если для каких-то версий оно не выполняется, то происходит удаление файлов по пути и удаление этих версий из массива.
7. Обновляются данные в массиве Files через смарт-контракт, фиксируется состояние новой транзакции и происходит обновление всех узлов DLT.

2. Расчет надежности

Для оценки надежности предлагаемой концептуальной архитектуры с использованием стирающего кода нужно провести расчет схемы надежности [9].

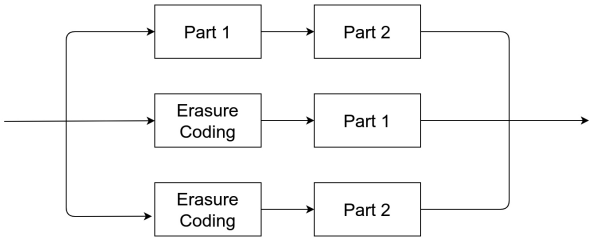

Рис. 2. Надежность предлагаемой системы

Каждый элемент нашей системы (компонент) расположен на отдельном сервере. Таким образом, файл можно получить одним из следующих способов (рис. 2):

- Через части 1 и части 2 файла.
- Через Erasure coding и части 1файла.
- Через Erasure coding и части 2 файла.

Случай «ИЛИ» вычисляется следующим образом:

$R= 1-[(1-R_1)(1-R_2)...(1-R_n)]$ (где R – надежность каждого компонента параллельно).

Случай «И»:

$$R= R_1*R_2*\ldots*R_n .$$

Если у нас есть случай, показанный на рис. 3.

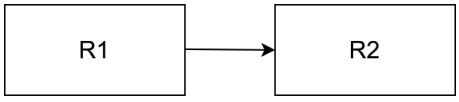

Рис. 3. Случай «И» для одинаковых компонент

Это случай «И», для R_1 и R_1. Тем самым, $R = R_1^2$. Так как это тот же компонент, то R_1^2 может быть заменен на R_1.

Таким образом, общая надежность может быть рассчитана следующим образом:

$$R=1 - (1 - R_1 * R_2) * (1 - R_1 * R_3) * (1 - R_2 * R_3) =$$
$$= 1 - (1 - R_2 * R_3 - R_1 * R_3 - R_1 * R_2 + R_1 * R2 * R_3^2 +$$
$$+ R_1 * R_2^2 * R_3 + R_1^2 * R_2 * R_3 - R_1^2 * R_2^2 * R_3^2) =$$
$$= R_2 * R_3 + R_1 * R_3 + R_1 * R_2 - 2 * R_1 * R_2 * R_3.$$

Если предположить, что каждый компонент имеет надежность 0,9, это означает, что общая надежность

$$R = 0.9*0.9+0.9*0.9+0.9*0.9-2*0.9*0.9*0.9 = 0.972$$

Решение по обеспечении надежности на основе полного дублирования всех компонент нашей системы безусловно является наиболее надежным способом (рис. 4). При этом оно не является экономически эффективным (если для развертывания системы требуется 8 серверов, значит, еще 8 серверов требуются для резервного копирования).

В случае общая надежность одного компонента будет равна:

$$R = 1-(1-R_1)(1-R_2) = 0.99.$$

Расчет надежности в схеме без серверов резервного копирования:

$$R = R_1*R_2 = 0.81.$$

Расчет надежности с полным дублированием:

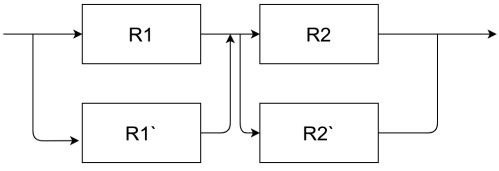

Рис. 4. Полное резервное копирование централизованной системы

$$R=[1-(1-R_1)(1-R_1`)]*[1-(1-R_2)(1-R_2`)] = 0.9801.$$

Предложенный подход имеет надежность 0.972 вместо 0.9801 (в случае полного дублирования), но при этом общий размер необходимого места будет на 25% меньше.

3. Обсуждение

Существуют другие решения, сочетающие технологии DHT и распределенные файловые хранилища, такие как BitTorrent file system [10] и Storj [12]. Оба этих проекта нацелены на конечных пользователей (B2C). Один пользователь запрашивает сохранение своих данных на компьютерах других пользователей. Для этого пользователь должен платить тем, кто размещает его данные, используя криптовалюту (токен Storj или BT). Рассмотренный в статье подход ориентирован на компании и позволяет им хранить файлы распределенным и надежным способом без использования криптовалюты.

Многочисленные торренты, построенные на P2P, работают следующим образом. Если узел запрашивает файл, то этот одноранговый узел делает копию файла и может поделиться ею. Таким образом, можно иметь файл с тысячами копий (поскольку тысячи узлов запросили один и тот же файл). Одновременно с этим нет контроля доступности узлов и контроля удаленных файлов, т.е. может сложится ситуация, что файл больше недоступен. Такое решение нельзя считать надежным B2B решением .

Рассматриваемые здесь идеи похожи на принцип работы технологии CDN [13]. CDN широко применяется для загрузки файлов за счет размещения наиболее часто используемых файлов (которые определяются в основном с использованием алгоритма LRU) на разных серверах в разных географических точках. По этой причине CDN можно считать более дорогим решением, которое способствует скорости загрузки файла браузерами, а не ограничению размера хранилища.

В качестве файлового хранилища для нашей системы можно использовать IPFS [14], который рассматривается многими исследователями как перспективное дополнение к DHT [3]. Но IPFS построен на тех же принципах, что и torrent решения и не подходят компаниям с точки зрения пропускной способности сети и отсутствия методов резервного копирования.

Заключение

Подведем итог:

- Рассмотрена архитектура хранения данных электронных документов в DHT и распределенной файловой структуре. Обсуждаются основные плюсы и минусы такой архитектуры и сравниваются с существующими решениями.
- Предложен метод хранения файлов в DLT и P2P, который уменьшает место хранения данных на 25% с использованием Erasure coding.
- Расчет надежности такой системы показывает, что в целом она более надежна чем централизованная и не существенно уступает системам с полным дублированием основных компонентов.

В статье изложена лишь концепция данной методологии. Для создания надежной и безопасной системы необходимы дополнительные функции шифрования файлов и блоков, проверки их целостности и протоколов передачи данных.

Для реализации подхода наиболее подходящим блокчейн-решением видится Ethereum Enterprise по следующим причинам:

- Платформа предназначена для создания корпоративного решения по модели B2C.
- Нет необходимости в использовании криптовалюты.
- Наличие смарт-контрактов и языка их программирования Solidity.
- Он поддерживает конфиденциальные транзакции (возможность настроить доступ к определенным транзакциям для определенных пользователей).

Нужно отметить, что рассмотренная методология хранения DApps легко масштабируется горизонтально в части распределенного файлового хранилища, если рассматривать всю систему как множество кластеров по 3 сервера (два главных и один для хранения стирающего кода).

Литература

1. *Капарулин П.А., Горелова Т.П.* Тенденции развития электронного документооборота в России //Вестник Академии. 2018. №. 2. С. 63-70.
2. *Kemp B., Olivan J.* European data format ‘plus’ (EDF+), an EDF alike standard format for the exchange of physiological data // Clinical Neurophysiology. 2003. Том 114. Выпуск 9. P. 1755-1761.
3. *Golosova J., Romanovs A.* The Advantages and Disadvantages of the Blockchain Technology // IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE). Литва. 2018
4. *Wu K. et al.* A first look at blockchain-based decentralized applications //Software: Practice and Experience. 2019.

5. *Johnston D. et al.* The General Theory of Decentralized Applications, DApps, URL- https://cryptochainuni.com/wp-content/uploads/The-General-Theory-of-Decentralized-Applications-DApps.pdf – 2014.
6. *Yongle Chen, Hui Li , Kejiao Li, Jiyang Zhang.* An improved P2P File System Scheme based on IPFS and Blockchain // 2017 IEEE International Conference on Big Data (Big Data). США. 2017.
7. *Соловьев А.В., Тарханов И.А.* Электронные документы и задача обеспечения сохранности при обмене данными в цифровой экономике //Труды Института системного анализа РАН. 2018. Т. 68. №. 1. С. 42-53.
8. *Balaji S.B. et al.* Erasure coding for distributed storage: An overview //Science China Information Sciences. 2018. Т. 61. №. 10. С. 1-45.
9. *Menčík J.* Reliability of Systems //Concise Reliability for Engineers. 2016. С. 33.
10. *Xu J., Figueiredo R.* Gatorshare: a file system framework for high-throughput data management //Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. 2010. С. 776-786.
11. *Jiaxing Li a , Jigang Wua,b, Long Chen.* Block-secure: Blockchain based scheme for secure P2P cloud storage // Information Sciences. 2018. № 465.P. 219-231
12. *Storj Labs*, Inc. Storj. A Decentralized Cloud Storage Network Framework. 2018.
13. *Gang Peng.* CDN: Content Distribution Network // arXiv. № 0411069. 2004.
14. *Juan Benet.* IPFS – Content Addressed, Versioned, P2P File System // arXiv. № 1407.3561v1. 2014.

Тарханов Иван Александрович. Федеральное государственное учреждение «Федеральный исследовательский центр «Информатика и управление» Российской академии наук», г. Москва, Россия. Старший научный сотрудник, кандидат технических наук, доцент. Количество печатных работ: 37. Область научных интересов: электронный документооборот, блокчейн, информационная безопасность.
E-mail: tarkhanov@isa.ru

Хаммуд Обадах. Национальный исследовательский технологический университет (НИТУ) «МИСиС», г. Москва, Россия. Научный ассистент. Количество печатных работ: 4. Область научных интересов: Dapps, блокчейн. E-mail: obadah.hammoud@gmail.com

Methodology for storing electronic documents in decentralized applications on the blockchain (DApps)

O. Hammoud[I], I.A. Tarkhanov[II,III]

[I] National University of Science and Technology “MISiS”, Moscow, Russia

[II] State Academic University for Humanities, Moscow, Russia

[III] Federal Research Center “Computer Science and Control” of Russian Academy of Sciences, Moscow, Russia

Abstract. This paper examines the problem of distributed storage of metadata for electronic documents in decentralized blockchain-based systems (DApps). Also, known ways of storing files in distributed systems, including central storage, torrents, IPFS, and Storj, are discussed. An approach is proposed for constructing such systems that allows to optimize the size of the space required for storage and at the same time ensure secure storage of data in geographically distributed systems of a company, or within a consortium of companies. The reliability of this approach is calculated, and the result is compared with the classic solution based on full backup.
Keywords: *blockchain, p2p, distributed file storage, electronic document, reliability*
DOI: 10.14357/20790279210205

References

1. *Kaparulin P.A., Gorelova T.P.* Tendencii razvitija jelektronnogo dokumentooborota v Rossii //Vestnik Akademii. 2018. No. 2. P. 63-70.
2. *Kemp B., Olivan J.* European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data // Clinical Neurophysiology. 2003. Vol. 114. Issue 9. P. 1755-1761.
3. *Golosova J., Romanovs A.* The Advantages and Disadvantages of the Blockchain Technology // IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Lithuania. 2018.
4. *Wu K. et al.* A first look at blockchain-based decentralized applications //Software: Practice and Experience. 2019.
5. *Johnston D. et al.* The General Theory of Decentralized Applications, DApps, URL- https://cryptochainuni.com/wp-content/uploads/The-General-Theory-of-Decentralized-Applications-DApps.pdf 2014.
6. *Yongle Chen, Hui Li, Kejiao Li, Jiyang Zhang.* An improved P2P File System Scheme based on IPFS and Blockchain // 2017 IEEE International Conference on Big Data (Big Data). США, 2017.
7. S*olovjev A.V., Tarkhanov I.A.* Jelektronnye dokumenty i zadacha obespechenija sohrannosti pri obmene dannymi v cifrovoj jekonomike //Trudy Instituta sistemnogo analiza Rossijskoj akademii nauk. 2018. Vol. 68. No. 1. P. 42-53.
8. *Balaji S.B. et al.* Erasure coding for distributed storage: An overview //Science China Information Sciences. 2018. Vol. 61. No. 10. P. 1-45.
9. *Menčík J.* Reliability of Systems //Concise Reliability for Engineers. 2016. P. 33.
10. *Xu J., Figueiredo R.* Gatorshare: a file system framework for high-throughput data management //Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing. 2010. P. 776-786.
11. *Jiaxing Li a , Jigang Wua,b, Long Chen*. Block-secure: Blockchain based scheme for secure P2P cloud storage // Information Sciences. 2018. No. 465. P. 219-231
12. Storj Labs, Inc. Storj. A Decentralized Cloud Storage Network Framework. 2018.
13. *Gang Peng.* CDN: Content Distribution Network // arXiv. No. 0411069. 2004.
14. *Juan Benet.* IPFS – Content Addressed, Versioned, P2P File System // arXiv. № 1407.3561v1. 2014.

Tarkhanov I.A. PhD, Institute for Systems Analysis Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Moscow, Russia/ E-mail: tarkhanov@isa.ru

Hammoud Obadah. Assistant, National University of Science and Technology "MISiS", Moscow, Russia. E-mail: obadah.hammoud@gmail.com