http://dx.doi.org/10.35596/1729-7648-2024-30-3-61-68

Оригинальная статья
Original paper

УДК 339.138; 004.7.371

ИСПОЛЬЗОВАНИЕ БЛОКЧЕЙНА ETHEREUM В СЕТИ ИНТЕРНЕТА ВЕЩЕЙ ДЛЯ IT-ДИАГНОСТИКИ

В. А. ВИШНЯКОВ, ИВЭЙ СЯ, ЧУЮЭ ЮЙ

Белорусский государственный университет информатики и радиоэлектроники
(г. Минск, Республика Беларусь)

Поступила в редакцию 23.01.2024

Аннотация. В статье рассмотрено использование технологии блокчейн Ethereum в сети интернета вещей (IoT) для IT-диагностики пациентов, что повышает безопасность данных и конфиденциальность пользователей. Такая интеграция оказывается эффективной для хранения и управления конфиденциальными данными пациентов с неврологическими болезнями. Разработана архитектура интегрированной системы, которая объединяет сеть IoT, файловую структуру IPFS (InterPlanetary File System) с блокчейном Ethereum для создания надежной модели хранения данных. Эта система обеспечивает эффективную, безопасную и прозрачную обработку данных, оптимизируя процессы их регистрации, авторизации и проверки. Использование IPFS для децентрализованного хранения файлов, наряду с блокчейном Ethereum, с целью создания защищенных от несанкционированного доступа медицинских записей обеспечивает повышение эффективности, масштабируемости и конфиденциальности. При проведении экспериментов реализован процесс создания и тестирования системы, включая настройку среды, подключение узла IPFS, программирование смарт-контрактов Ethereum, выборку голосовых данных и хранение их хэшей.

Ключевые слова: интернет вещей, блокчейн Ethereum, файловая система, голосовые данные, конфиденциальность.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Для цитирования. Вишняков, В. А. Использование блокчейна Ethereum в сети интернета вещей для IT-диагностики / В. А. Вишняков, Ивэй Ся, Чуюэ Юй // Цифровая трансформация. 2024. Т. 30, № 3. С. 61–68. http://dx.doi.org/10.35596/1729-7648-2024-30-3-61-68.

USING THE ETHEREUM BLOCKCHAIN IN THE INTERNET OF THINGS NETWORK FOR IT DIAGNOSTICS

ULADZIMIR A. VISHNIAKOU, YIWEI XIA, CHUYUE YU

Belarusian State University of Informatics and Radioelectronics (Minsk, Republic of Belarus)

Submitted 23.01.2024

Abstract. The article discusses the use of Ethereum blockchain technology in the Internet of Things (IoT) network for IT diagnostics of patients, which increases data security and user privacy. This integration is proving effective for storing and managing sensitive data of patients with neurological diseases. An integrated system architecture has been developed that combines the IoT network, the IPFS (InterPlanetary File System) file structure with the Ethereum blockchain to create a reliable data storage model. This system ensures efficient, secure and transparent data processing, optimizing the processes of data registration, authorization and verification. Using IPFS for decentralized file storage, along with the Ethereum blockchain to create tamper-proof medical records, provides increased efficiency, scalability and privacy. During the experiments, the process of creating and testing the system was implemented, including setting up the environment, connecting an IPFS node, programming Ethereum smart contracts, sampling voice data and storing their hashes.

Keywords: Internet of things, Ethereum blockchain, file system, voice data, privacy.

Conflict of interests. The authors declare no conflict of interest.

For citation. Vishniakou U. A., YiWei Xia, Chuyue Yu (2024) Using the Ethereum Blockchain in the Internet of Things Network for IT Diagnostics. *Digital Transformation.* 30 (3), 61–68. http://dx.doi.org/10.35596/1729-7648-2024-30-3-61-68 (in Russian).

Введение

Технология интернета вещей (IoT) [1, 2] включает сеть устройств и датчиков, соединенных между собой через интернет, способных собирать, обмениваться и обрабатывать данные для обеспечения эффективных и интеллектуальных операций. Однако технология IoT имеет проблемы в области безопасности данных, защиты конфиденциальности, которые варьируются от рисков утечки данных, уязвимостей в безопасности устройств и данных. Для обеспечения безопасности среды интернета вещей и эффективной защиты данных пользователей необходима стратегия безопасности, включающая усиление защиты устройств, шифрование данных, разработку и обеспечение соблюдения политик защиты конфиденциальности пользователей.

Технология блокчейн Ethereum [3] обеспечивает безопасность и неизменность данных благодаря хэшированию, шифрованию и децентрализации. Ее применение в сетях IoT позволит обеспечить безопасность данных и конфиденциальность пользователей [4]. Ethereum – это платформа смарт-контрактов, основанная на технологии блокчейн, используемая для создания децентрализованных приложений (DApps). Смарт-контракты, развернутые в Ethereum, представляют собой распределенные программы, работающие на нескольких узлах сети. Когда пользователи сохраняют хэш-значение в смарт-контракте с помощью его функций, хэш-значение записывается в состояние каждого узла сети Ethereum, обеспечивая неизменность данных. Одновременно в цепочке может быть установлена индексация для удобства поиска, реализуемая путем записи функций поиска в контракт. Способность смарт-контрактов в рамках технологии Ethereum автоматизировать выполнение транзакций повышает эффективность управления данными.

В статье рассмотрено использование технологии Ethereum в сетях IoT IT-диагностики пациентов при хранении медицинских диагностических данных с болезнью Паркинсона [5] и Альцгеймера [6].

Структура хранения данных IoT с использованием файловой системы и Ethereum

Как показано на рис. 1, в системе интернета вещей IT-диагностики данные от пациентов, оснащенных датчиками, отправляются на локальный сервер [7]. Сервер получает данные по протоколу HTTP и взаимодействует с веб-сервером с использованием WSGI (Web Server Gateway Interface). Сервер использует программу Flask для обработки данных, которые затем обрабатываются моделями нейронных сетей GRU, LSTM и RNN. Эти модели анализируют данные для генерации прогнозов. Сервер также включает базу данных для их хранения и извлечения.

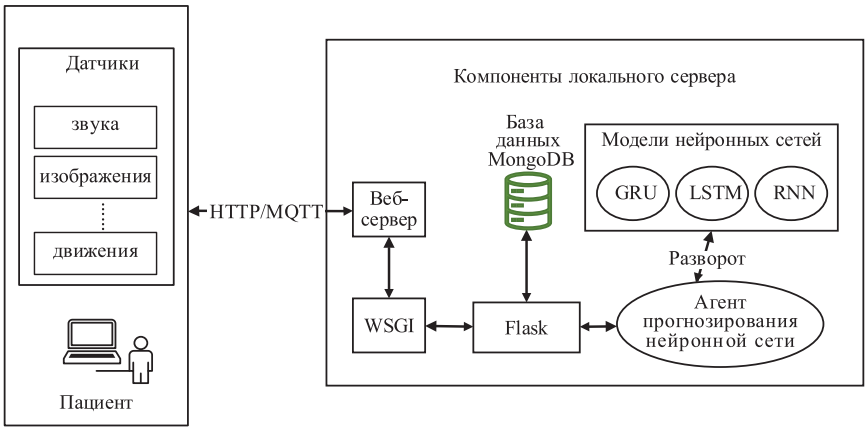

Рис. 1. Архитектура системы интернета вещей на основе нейронной сети
Fig. 1. Architecture of an Internet of things system based on a neural network

На рис. 2 показана разработанная структура системы хранения данных IoT, использующая технологию блокчейн Ethereum и включающая процесс регистрации, авторизации и проверки данных через Ethereum и файловую систему (IPFS).

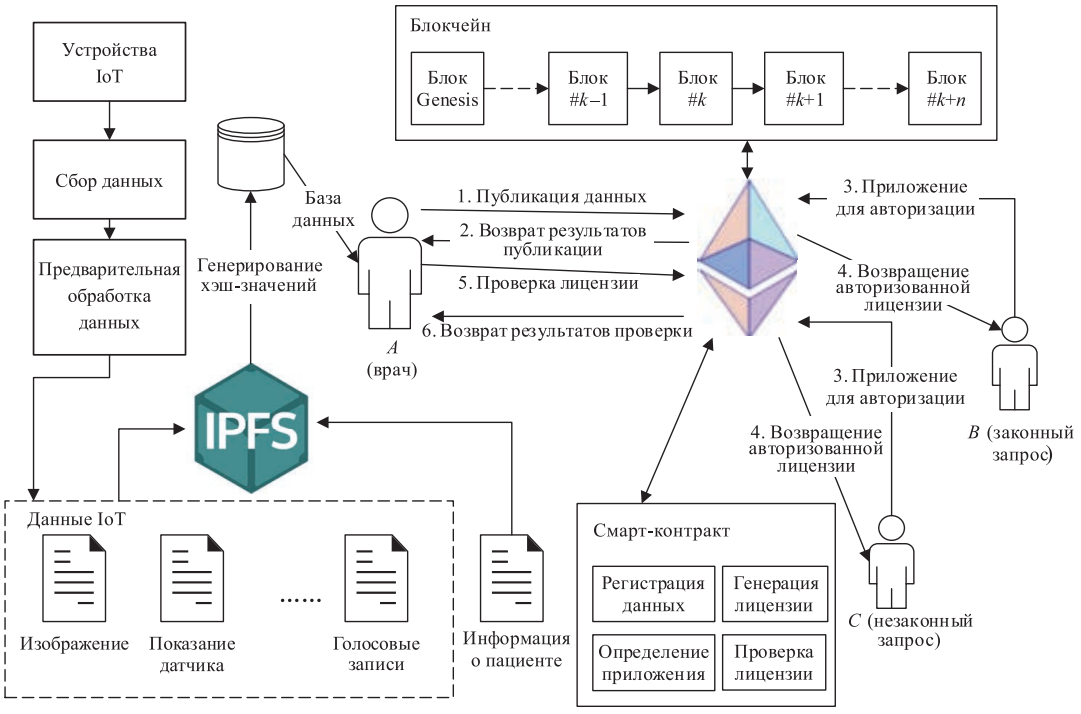

Рис. 2. Структура хранения данных с использованием Ethereum и IPFS
Fig. 2. Data storage structure using Ethereum and IPFS

Основные компоненты, изображенные на рис. 2, включают:

– сеть блокчейн: начинается с блока Genesis, за которым идет последовательность взаимосвязанных блоков (#*k*–1, #*k*, #*k*+1, #*k*+*n*). Это – структура данных блокчейна, где каждый блок содержит ряд транзакций, связанных через хэш-значения с предыдущим блоком;

– взаимодействие с пользователем: пользователь *A* отвечает за загрузку в систему данных, которые могут быть изображениями, показаниями датчиков, голосовыми записями или информацией о пациенте. Пользователи *B* и *C* запрашивают авторизацию для доступа к определенным данным или выполнения действий. Пользователь *B* получает разрешение на предоставление лицензии, в то время как заявка пользователя *C* завершается неудачей из-за незаконного запроса (например, из-за отсутствия лицензии);

– у смарт-контрактов – ключевая роль в системе, они выполняют регистрацию данных, генерацию лицензий, определение приложения и проверку лицензий.

Рассмотрим алгоритм работы системы IoT.

1. Ввод данных пациента и сбор данных с устройств IoT: пациенты используют смартфоны и различные устройства IoT для входа в систему здравоохранения и отправки своих личных медицинских данных. Эти устройства могут быть носимыми простыми датчиками, отслеживающими частоту сердечных сокращений и уровни активности, либо сложными датчиками, которые отслеживают уровень глюкозы в крови или кровяное давление.

2. Загрузка и хранение данных: данные загружаются на локальный сервер Flask (рис. 1) с использованием протокола HTTP. Сервер Flask обрабатывает запрос и сохраняет полученные показатели в базе данных MongoDB, которая действует как центральное хранилище для этой информации.

3. Обработка прогнозирования заболеваний: сервер Flask обрабатывает сохраненные данные, отправляя их агенту прогнозирования нейронной сети (рис. 1). Прогнозы, сделанные этим агентом, сохраняются обратно в базу данных MongoDB для ведения записей и дальнейшего использования.

4. Распространение результатов среди клиентов: пациенты получают результаты прогнозирования через HTTP-ответ. Кроме того, результаты распространяются среди врачей в режиме реального времени по протоколу MQTT, поддерживаемому брокером EMQX.

5. Резервное копирование данных через IPFS: в сеть IPFS экспортируется база данных MongoDB, которая включает как необработанные данные о пациентах, так и сгенерированные прогнозы.

6. Запись хэш-значений в блокчейн: всякий раз, когда резервная копия базы данных создается из MongoDB и сохраняется в IPFS, результирующие хэш-значения фиксируются в блокчейне. Пользователь *A*, который авторизован для обработки данных пациента, инициирует этот процесс. База данных MongoDB извлекает существующие медицинские данные пациента из IPFS всякий раз, когда возникает необходимость записать новые данные для пациента.

7. Контроль доступа к данным смарт-контрактов. Смарт-контракты на блокчейне запрограммированы для осуществления критически важных функций в управлении медицинскими данными и доступе к ним. Они выполняют:

– регистрацию данных: когда пользователь *A* загружает новые данные в IPFS и записывает соответствующие хэш-значения в блокчейн, смарт-контракт регистрирует эти записи, гарантируя происхождение данных;

– генерацию лицензии: по запросу пользователя *B* смарт-контракт генерирует лицензию или токен, который предоставляет доступ к указанным данным. Этот процесс включает проверку учетных данных пользователя *B* и его намерений обеспечить соответствие политике доступа к данным;

– определение приложения: смарт-контракт автоматизирует оценку запросов на доступ. Когда пользователь *B* подает заявку на доступ к данным, смарт-контракт определяет законность запроса на основе предопределенных правил. И, наоборот, если пользователь *C* делает запрос, смарт-контракт идентифицирует его, как несанкционированный (из-за отсутствия предопределенных правил), и отказывает в доступе;

– проверку лицензии: при попытке доступа к данным смарт-контракт проверяет соответствие выданным лицензиям или токенам. Доступ к данным, хранящимся в IPFS, предоставляется только запросам, в которых указана лицензия, например, выданная пользователю *B*. Пользователю *C*, не имеющему такой лицензии, будет отказано, что гарантирует сохранность данных от несанкционированных изменений или взломов.

В контексте IT-диагностики пациентов медицинский работник (врач) (пользователь *A*) загружает информацию о пациенте, которая может включать медицинские записи или медицинские изображения. Смарт-контракты играют решающую роль в этой системе, управляя регистрацией данных в блокчейне, обеспечивая их подлинность с помощью уникальных значений хэша. Они также обрабатывают выдачу лицензий на доступ к данным, оценивая и подтверждая запросы пользователей на основе предопределенных критериев. Например, врач, запрашивающий доступ к медицинской карте пациента, получит лицензию после выполнения необходимых условий. Файловая система IPFS [8] используется для децентрализованного хранения медицинских файлов, что повышает безопасность и доступность данных. Система гарантирует, что только авторизованный персонал может получить доступ к конфиденциальным медицинским данным, тем самым сохраняя конфиденциальность и целостность при управлении медицинскими данными.

Смарт-контракт разработан с целью обеспечения безопасной структуры для хранения, управления и обмена медицинской информацией. В его состав входят:

– структуры данных: хранят хэш-значение медицинских записей, адрес владельца и статус доступности записи. Управляют правами доступа к медицинским записям для конкретных лиц (например, врачей, исследователей);

– сопоставление хэшей данных с соответствующими медицинскими записями: сопоставляются хэши данных и индивидуальные адреса с их правами доступа;

– события для регистрации медицинской карты, предоставления, отзыва и проверки разрешений на доступ.

Интеграция IPFS с технологией блокчейн Ethereum поддерживает хранение больших данных, предлагая безопасное и эффективное решение, подходящее для управления данными здравоохранения и пациентов. Система повышает надежность и долговечность данных за счет снижения зависимости от централизованных серверов благодаря IPFS, децентрализованной системе хранения. Это обеспечивает целостность и неизменяемость данных. Изменения в данных изменяют их хэш-значение IPFS, что облегчает проверку. Хэш-значения, хранящиеся в блокчейне, обеспечивают постоянную запись, используя блокчейн.

Предложенный подход превосходит традиционное облачное хранилище по экономической эффективности, особенно для больших данных. Кроме того, он повышает безопасность данных, шифруя их в IPFS и сохраняя только хэш-значения в блокчейне, таким образом защищая конфиденциальную информацию. Эта технология применяется в управлении медицинскими записями, обеспечивая неизменность и целостность данных.

Пользователь *A* публикует данные, которые затем отправляются в IPFS и регистрируются в блокчейне, создавая хэш-значение. Пользователи *B* и *C* обращаются к смарт-контракту для сбора или проверки данных, и смарт-контракт определяет, предоставлять ли авторизацию на основе встроенных правил. Для авторизованных запросов смарт-контракт генерирует лицензию и возвращает ее пользователю, для неавторизованных – возвращает результат сбоя.

Работа с системой, эксперименты

Настройка среды: установка и инициализация Node.js, версия 16.13.0 [9].

Создание узла IPFS и загрузка голосовых файлов. Пользователи могут включать неизменяемые и постоянные ссылки доступа из IPFS в транзакции блокчейна. Последовательность такова.

1. После загрузки и установки IPFS Desktop появится его стартовая страница, как показано на рис. 3. На этом этапе узел IPFS установлен и работает на компьютере; при запуске IPFS Desktop он автоматически инициирует узел IPFS, позволяя обмениваться данными с другими узлами IPFS. Далее используется метод взаимодействия на основе сценариев для добавления голосового файла в сеть IPFS через API, обеспечивая загрузку голосового файла и получение сгенерированного значения хэша IPFS.

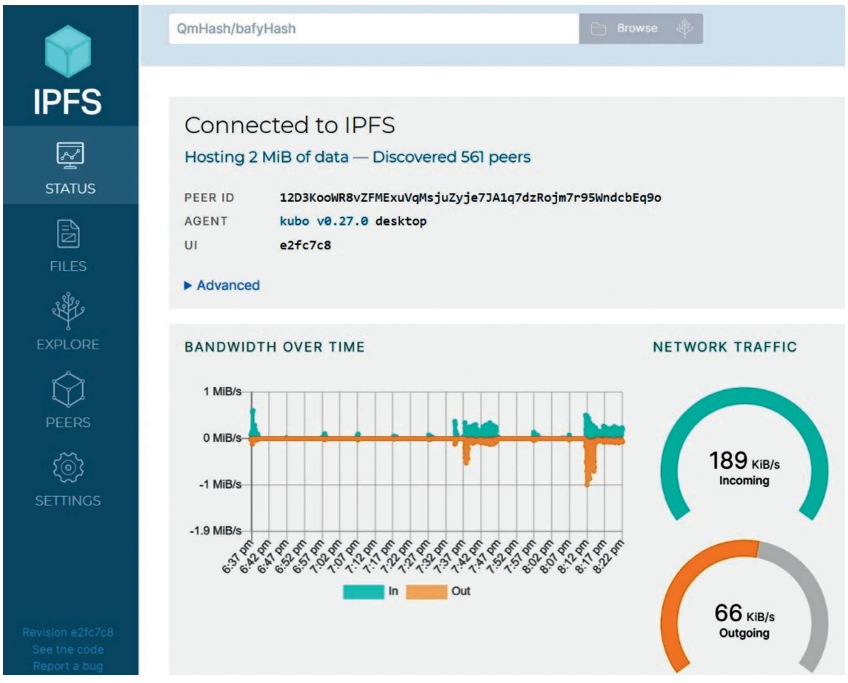

Рис. 3. Главная страница IPFS Desktop
Fig. 3. IPFS Desktop home page

2. Подготовка голосового файла с расширением .wav и запись пути к его хранилищу.

3. Написание скрипта для установления соединения с локальным узлом IPFS, применяя клиентскую библиотеку Kubo RPC. Использование ее API для операций добавления файлов (хранения), который возвращает уникальное значение хэша IPFS. Значение хэша файла, хранящегося в IPFS, может быть возвращено с помощью инструкции `return file.cid.toString();`. Как только файл добавлен в сеть IPFS и получено его значение хэша, другие пользователи могут извлекать, загружать или получать доступ к содержимому того же файла, используя это значение хэша.

Хранение хэш-значения в сети Ethereum. В эксперименте использовалась платформа разработки Truffle Ethereum для развертывания контрактов. Последовательность такова.

1. Установка Truffle [10]. После установки Node.js пользователи могут установить Truffle, выполнив следующую команду в терминале с помощью диспетчера пакетов Node (npm):

`npm install truffle@5.1.1.5`.

2. Создание нового проекта Truffle. Инициализация Truffle: после установки Truffle открывается новый терминал и выполняются следующие команды:

`mkdir AudioStorageProject`

`cd AudioStorageProject`

`truffle init`.

3. Написание смарт-контракта для хранения и извлечения хэш-значений. Написание кода контракта, который определяет процесс развертывания смарт-контракта, включая компиляцию и развертывание в целевой сети.

4. Настройка Truffle. Отредактировать файл truffle-config.js для подключения к локальной сети.

5. Скомпилировать смарт-контракт. Запустить следующую команду в корневом каталоге проекта: `truffle compile`.

6. Развернуть контракт в локальной сети с помощью Truffle.

Взаимодействие с использованием смарт-контрактов Web3.js. Получив значение хэша файла и успешно развернув смарт-контракт с возможностью хранения данных в сети Ethereum, можно взаимодействовать со смарт-контрактом, используя Web3.js для сохранения значения хэша голосового файла в блокчейне Ethereum. Web3.js является стандартной библиотекой JavaScript для работы с Ethereum, позволяющей разработчикам взаимодействовать с блокчейном Ethereum напрямую из веб-приложений.

Код `const web3 = new Web3('http://127.0.0.1:7545');` в Web3.js используется для подключения к узлу Ethereum через его URL. Здесь 127.0.0.1 представляет локальный хост (localhost), а 7545 – номер порта узла Ethereum. Этот номер порта можно найти в программном обеспечении Ganache, как показано на рис. 4. Ganache служит локальной тестовой сетью для Ethereum, предоставляя имитацию блокчейна Ethereum, где 7545 является портом прослушивания по умолчанию для Ganache.

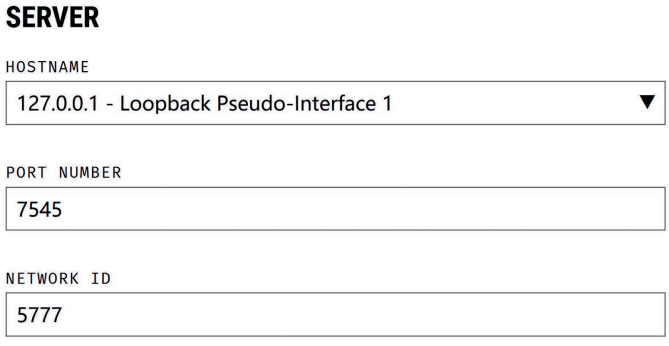

Рис. 4. Отображение порта по умолчанию для узла Ethereum
Fig. 4. Display of the default port for the Ethereum node

В коде `const contract Address = `0x316044A586191b9DEcA8 e9AC2AF1b3fF9 cc646Eb`;` адрес контракта относится к адресу развертывания смарт-контракта в сети Ethereum. Каждый смарт-контракт при развертывании в сети Ethereum получает уникальный адрес. Этот адрес действует как идентификатор смарт-контракта в Ethereum. При входе в Ganache «адрес контракта», отображаемый в Ganache, представляет собой адрес развертывания смарт-контракта в тестовой цепочке Ethereum (смоделированный в Ganache). Данный адрес присваивается во время развертывания смарт-контракта тестовой цепочке с помощью инструмента развертывания Truffle, как показано на рис. 5.

Последовательность действий такова.

1. Используется скрипт для создания экземпляра Web3, который подключается к локальному узлу Ganache.

2. Запуск Ganache.

3. Выбор «быстрый запуск», и программное обеспечение автоматически сгенерирует десять учетных записей Ethereum. Выберите любую учетную запись для выполнения транзакций.

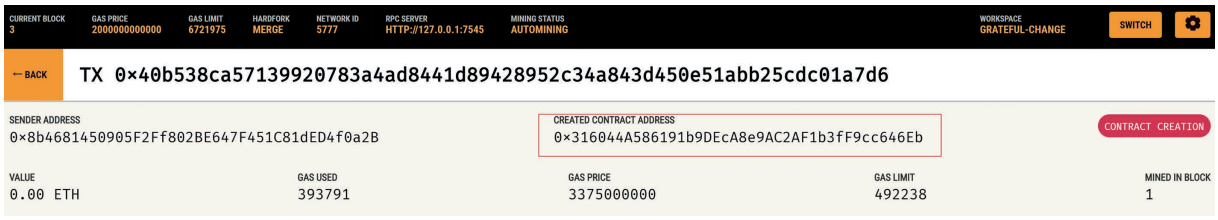

Рис. 5. Адрес развертывания смарт-контракта
Fig. 5. Smart contract deployment address

4. Запускается файл скрипта, ответственный за хранение аудиоданных.

5. Открыть программное обеспечение Ganache, чтобы проверить, было ли аудио успешно сохранено в сети Ethereum. Как показано на рис. 6, в интерфейсе Ganache появляется новая транзакция. Хэш транзакции (tx hash) представляет собой уникальный идентификатор для этой транзакции, используемый для ее отслеживания и подтверждения.

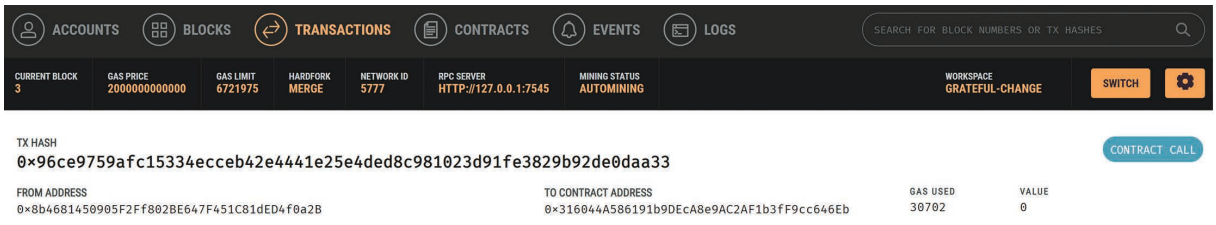

Рис. 6. Новая информация о транзакции
Fig. 6. New transaction information

Кроме того, можно наблюдать общее количество транзакций между ранее выбранным адресом учетной записи Ethereum и смарт-контрактом. Как показано на рис. 7, конкретная учетная запись Ethereum совершила три транзакции со смарт-контрактом, при этом количество транзакций (tx count) равно 3.

Рис. 7. Количество транзакций учетной записи Ethereum
Fig. 7. The number of transactions of the Ethereum account

Извлечение и доступ к голосовым данным. Успешно сохранив значение хэша файла в сети Ethereum и восстановив его, переходим к доступу к голосовым данным через это значение хэша следующим образом:

– извлечь хэш-значение IPFS желаемого аудио из сети Ethereum с помощью смарт-контракта;

– получить доступ к голосовым данным через хэш-значение IPFS.

Заключение

1. Реализована интеграция блокчейна Ethereum и файловой системы IPFS с сетью IoT IT-диагностики для создания конфиденциальности хранения больших данных пациентов.

2. Разработана система хранения данных IoT с использованием блокчейна Ethereum в сети IT-диагностики, позволяющая медицинским работникам безопасно загружать информацию о пациентах. Смарт-контракты управляют аутентичностью данных и контролем доступа.

3. Представлена интеграция файловой системы IPFS с технологией Ethereum, что повысило безопасность данных (за счет хэшей), обеспечив альтернативу традиционному облачному хранилищу, особенно для крупномасштабного объема данных.

4. Описан процесс создания и тестирования системы, включая настройку среды, подключение узла IPFS, программирование смарт-контрактов Ethereum, поиск голосовых данных и доступ к ним. Это создает основу для практического применения системы при управлении конфиденциальными медицинскими данными и записями.

Список литературы / References

1. Sarker I. H., Khan A. I., Abushark Y. B. (2023) Internet of Things (IoT) Security Intelligence: A Comprehensive Overview, Machine Learning Solutions and Research Directions. *Mobile Networks and Applications*. 28 (1), 296–312.
2. Kumar M., Kumar A., Verma S. (2023) Healthcare Internet of Things (H-IoT): Current Trends, Future Prospects, Applications, Challenges, and Security Issues. *Electronics*. 12 (9).
3. Tikhomirov S. (2018) Ethereum: State of Knowledge and Research Perspectives. *Foundations and Practice of Security: 10th International Symposium*. 206–221.
4. Bahga A., Madisetti V. K. (2016) Blockchain Platform for Industrial Internet of Things. *Journal of Software Engineering and Applications*. 9 (10), 533–546.
5. Vishniakou U. A., YiWei Xia (2023) IT Diagnostics of Parkinson's Disease Based on the Analysis of Voice Markers and Machine Learning. *Doklady BGUIR*. 21 (3), 102–110. http://dx.doi.org/10.35596/1729-7648-2023-21-3-102-110.
6. Vishniakou U. A., YiWei Xia, Chuyue Yu (2023) Technology of Neurological Disease Recognition Using Gated Recurrent Unit Neural Network and Internet of Things. *Open Semantic Technologies for Intelligent Systems (OSTIS), Collection of Scientific Papers. Iss. 7.* Minsk, Belarusian State University of Informatics and Radioelectronics. 241–246.
7. Vishniakou U. A. (2023) *Specialized IoT Systems: Models, Structures, Algorithms, Hardware, Software Tools.* Minsk, Belarusian State University of Informatics and Radioelectronics (in Russian).
8. Trautwein D., Raman A., Tyson G. (2022) Design and Evaluation of IPFS: A Storage Layer for the Decentralized Web. *Proceedings of the ACM SIGCOMM*. 739–752.
9. *Node.js*. Available: https://www.npackd.org/p/org.nodejs.NodeJS/16.13 (Accessed 23 October 2023).
10. *Truffle*. Available: https://trufflesuite.com/ (Accessed 24 October 2023).

Вклад авторов

Вишняков В. А. выполнил постановку задачи, предложил концепцию интеграции, предоставил информацию о выбранной экспериментально платформе интернета вещей.

Ивэй Ся провел детализацию разработки.

Чуюэ Юй спланировала и выполнила эксперименты.

Author's contribution

Vishniakou U. A. completed the task statement, proposed the concept of integration, and provided information about the experimentally selected Internet of things platform.

YiWei Xia carried out the details of the development.

Chuyue Yu planned and executed the experiments.

Сведения об авторах

Вишняков В. А., д-р техн. наук, проф. каф. инфокоммуникационных технологий, Белорусский государственный университет информатики и радиоэлектроники (БГУИР)

Ивэй Ся, асп. каф. инфокоммуникационных технологий, БГУИР

Чуюэ Юй, асп. каф. инфокоммуникационных технологий, БГУИР

Information about the authors

Vishniakou U. A., Dr. of Sci. (Tech.), Professor at the Department of Infocommunication Technologies, Belarusian State University of Informatics and Radioelectronics (BSUIR)

YiWei Xia, Postgraduate at the Department of Infocommunication Technologies, BSUIR

Chuyue Yu, Postgraduate at the Department of Infocommunication Technologies, BSUIR

Адрес для корреспонденции

220013, Республика Беларусь,
г. Минск, ул. П. Бровки, 6
Белорусский государственный университет
информатики и радиоэлектроники
Тел.: +375 44 486-71-82
E-mail: vish@bsuir.by
Вишняков Владимир Анатольевич

Address for correspondence

220013, Republic of Belarus,
Minsk, P. Brovki St., 6
Belarusian State University
of Informatics and Radioelectronics
Tel.: +375 44 486-71-82
E-mail: vish@bsuir.by
Vishniakou Uladzimir Anatolievich