



Title	Secure Yet Practical PHR Sharing: A Hybrid Approach of NFT and Attribute-Based Encryption
Author(s)	Shijo, Yoshinobu; Miyanishi, Nanami; Ochiai, Shogo et al.
Citation	Digest of Technical Papers – IEEE International Conference on Consumer Electronics. 2026
Version Type	AM
URL	https://hdl.handle.net/11094/104909
rights	© 2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

Secure Yet Practical PHR Sharing: A Hybrid Approach of NFT and Attribute-Based Encryption

Yoshinobu Shijo*, Nanami Miyanishi*, Shogo Ochiai[†], Eiichiro Hashiba[†],
Miki Tani*, Masayuki Murata[‡], Kenji Yamada*

*Global Center for Medical Engineering and Informatics, The University of Osaka, Osaka, Japan

[†]Consortium for Advanced Kansai Medical Engineering and Informatics, Osaka, Japan

[‡]Graduate School of Information Science and Technology, The University of Osaka, Osaka, Japan

*{shijo.yoshinobu.mei, miyanishi.nanami.mei, tani.miki.mei, yamada.kenji.mei}@osaka-u.ac.jp

[‡]murata.masayuki.ist@osaka-u.ac.jp

Abstract—Personal Health Records (PHRs) enable personalized and continuous healthcare services, but contain highly sensitive information, requiring strong security and privacy safeguards. Self-sovereign architectures, where individuals retain full control over their data, represent a promising model for secure PHR sharing. In our prior work, we implemented a blockchain-based system using Non-Fungible Tokens (NFTs) to represent data ownership and usage rights. While NFTs provide tamper resistance, NFT-only access control is vulnerable to wallet compromise and requires explicit user consent, making it unsuitable for emergency access when patients are unconscious or otherwise unable to consent. To address these limitations, we newly propose a hybrid PHR-sharing framework combining NFTs with Attribute-Based Encryption (ABE). Our new approach enforces cryptographic access policies beyond NFT possession and enables emergency access to predefined medical information without explicit user consent. We analyze representative attack scenarios and show that the scheme provides secure access control and rights management. We implement a prototype and evaluate its performance. For 1 MB of data, used as a practical upper bound for text-based PHR records based on wearable-device measurements, retrieval takes approximately 1 second, while registration and access granting take approximately 12 and 6 seconds on the Base testnet, a high-speed Ethereum-compatible test network. These results demonstrate practical feasibility, with further optimization possible through faster blockchain networks or reduced blockchain transactions.

Index Terms—Personal Health Record (PHR), Non-Fungible Token (NFT), Attribute-Based Encryption (ABE), Emergency Access

I. INTRODUCTION

A Personal Health Record (PHR) comprises a wide variety of health information that an individual manages over a lifetime, including basic medical information such as past medical history and medication records; lifelog data such as sleep and diet logs; and automatically synchronized data obtained from wearable devices, such as heart rate and sleep records [1]. Leveraging PHRs enables access to services tailored to one's health condition and is expected to improve individual well-being [2]. However, PHRs contain highly sensitive information, requiring strong security measures. Conventional provider-centric models suffer from data silos and risks such as service discontinuation. As a result, the practical value of PHR data is reduced [3].

To address these issues, a self-sovereign architecture offers an effective solution. Here, “self-sovereign” refers to an architecture in which individuals have full control over their own data and can authorize third parties to access or use it selectively. Our research group has proposed a self-sovereign medical data management and distribution system that clarifies data ownership and usage rights using Non-Fungible Tokens (NFTs), and manages data circulation based on those rights [4], [5]. An NFT is a “digital certificate” issued on a blockchain with tamper resistance and is managed in the holder's blockchain wallet. This mechanism ensures tamper-resistant rights management and allows the system to identify the right holder through the associated blockchain wallet [6]. However, because rights and permission management rely solely on NFTs, data may be misused if a wallet is compromised or if an NFT is illicitly transferred via phishing or similar attacks. Moreover, in a self-sovereign architecture, explicit consent from the owner is required, which makes it difficult to utilize PHRs in emergencies where obtaining consent is infeasible.

Therefore, this paper proposes a self-sovereign PHR sharing system that leverages NFTs while enhancing security through a hybrid approach with Attribute-Based Encryption (ABE), and that remains applicable to practical PHR sharing scenarios. The contributions of this paper are as follows.

- 1) We enhance security, insufficient under NFT-only control, by employing Attribute-Based Encryption (ABE) so that only pre-registered users (e.g., physicians) can use the data.
- 2) We introduce a mechanism to obtain usage rights, without explicit consent, for a limited set of data categories, thereby enabling appropriate PHR utilization in emergencies.
- 3) We implement a prototype demonstrating practical performance, with data retrieval in about 1 s and acceptable registration/access times that can be further optimized.

The remainder of this paper is organized as follows: Section II presents practical use cases; Section III reviews related work; Section IV details our method; Section V provides

security verification; Section VI reports evaluation results; and Section VII concludes the paper.

II. PRACTICAL PHR SHARING USE CASES

We consider the following two practical PHR sharing use cases. The designed system must be capable of supporting these scenarios.

a) *Scenario 1*: A patient provides their PHR data to users such as physicians or researchers, who then utilize the data for diagnosis, treatment, or research and development. In this case, the patient specifies the recipient and grants data usage rights. The authorized user exercises the granted usage rights to access and use the data.

b) *Scenario 2*: In emergency situations, physicians need immediate access to a patient's PHR data in order to provide prompt treatment. In this case, physicians must be able to access the data without the patient's explicit consent. However, the accessible data must be strictly limited to essential information required in emergencies, such as allergy records and medical history [7]. Physicians obtain usage rights for this limited subset of data and use the data accordingly.

Note that patients can always access their own data. In addition, to prevent unauthorized access by third parties impersonating physicians or researchers, users are assumed to undergo a prior verification process and be registered in the system with appropriate attributes assigned.

III. RELATED WORK

A. Existing Works

Numerous methods for self-sovereign architectures of PHR management have been proposed in recent years. George et al. [8] and Kim et al. [9] present decentralized PHR management centered on Self-Sovereign Identity (SSI) and Decentralized Identifiers (DIDs). Klementi et al. [10] propose a Personal Health Data Space architecture based on decentralized content-addressable storage. Yakubu et al. [11] enforce access granting and auditing via smart contracts, positioning patients as the primary authority for permission management. These works pursue self-sovereign PHRs but do not support emergency access or address wallet-compromise threats.

B. Our Previous Works

We proposed a system that represents data ownership and usage rights with NFTs and manages data based on those rights [4], [5]; an overview is shown in Fig. 1(a). Among the use cases in Section II, this system satisfies Scenario 1; however, as indicated by P1 in Fig. 1(a), it does not satisfy Scenario 2. This is because usage rights are assumed to always require the owner's explicit consent, which is difficult to obtain in emergencies. Moreover, because rights and permissions are managed solely with NFTs, unauthorized transfer of an NFT by any means can lead to unauthorized data use (Fig. 1(a)-P2). Blockchain wallet security incidents have been increasing, making NFT-only rights management insufficient for sensitive PHRs. In addition, users can transfer NFTs to other users, which raises the risk of misuse (Fig. 1(a)-P3). As shown in

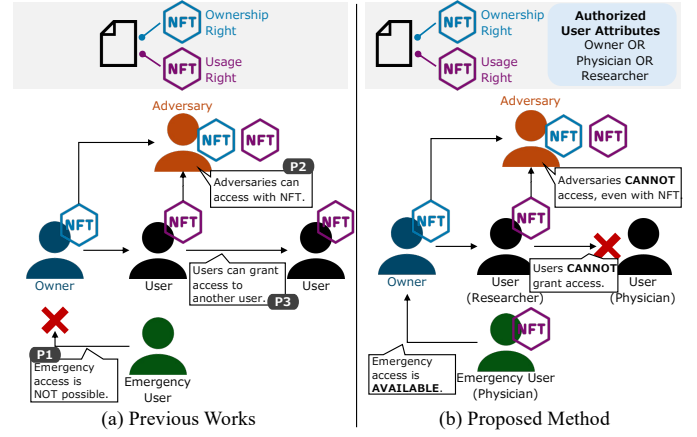


Fig. 1: Comparison of Previous Works and Proposed Method

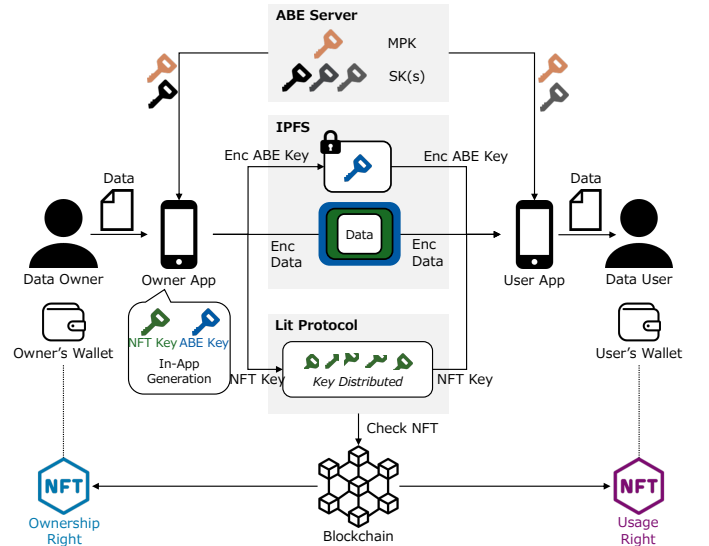


Fig. 2: System Overview

Fig. 1(b), this paper resolves these issues to realize a more secure yet practical PHR-sharing system.

IV. PROPOSED METHOD

The proposed method enhances security through a hybrid scheme combining NFTs and ABE, while also enabling the acquisition of usage-right NFTs without explicit consent for limited data categories. In doing so, it realizes practical PHR-sharing use cases. An overview of the proposed system is shown in Fig. 2.

A. Core Components

1) *NFT for Data Ownership and Usage Rights*: In the proposed method, NFTs represent both data ownership and data usage rights. An NFT is implemented as a smart contract deployed on a blockchain. We adopt ERC-721, which is widely used as the de facto standard interface specification, in our system.

2) *Lit Protocol for NFT-based Access Control*: Encrypted data access is controlled such that only users who hold the corresponding NFT can obtain the decryption key. To realize

NFT-based access control, we adopt Lit Protocol [12], a distributed key-management infrastructure composed of nodes called Lit Nodes. Keys managed by Lit Protocol to decrypt data are encrypted on the client side using the shared public key of Lit Protocol, and stored across Lit Nodes together with an Access Control Condition (ACC) that defines the decryption conditions. The secret key required to decrypt keys is split via Shamir's Secret Sharing. Let N be the number of Lit Nodes and t be the number of shares required for recovery; Lit Protocol sets $t \geq 2N/3$. Each Lit Node decides whether to release its share based on the ACC. The ACC can specify blockchain states, such as “user i holds the NFT,” as a condition. Access to Lit Protocol requires a signature generated by the user's blockchain wallet; Lit Nodes verify the signature to authenticate user i .

3) *Attribute-Based Encryption for Enhanced Security*: To further strengthen security, the proposed method introduces Attribute-Based Encryption (ABE). ABE decrypts data only if the user's attribute set satisfies the attribute policy associated with the ciphertext. Encryption of data $Data$ with public parameters MPK and allowed attribute set $Attr$ is performed as:

$$AbeEncData = AbeEnc(MPK, Data, Attr) \quad (1)$$

Decryption with an attribute key SK is performed as:

$$Data = AbeDec(SK, AbeEncData) \quad (2)$$

Each SK contains attributes S , and decryption succeeds only if S satisfies $Attr$. For example, if $Attr = A \vee (B \wedge C)$, then $S = \{A\}, \{B, C\}, \{A, B, C\}$ satisfy the condition, whereas other sets do not. In this method, available attributes are $Owner_i$ (a unique attribute for owner i), *Physician*, and *Researcher*.

MPK and SK are generated during an initial setup. A master secret key MSK is generated, and MPK and SK are derived from it. In the proposed method, an ABE Server performs this setup. Owners and users apply to the ABE Server and receive MPK and SK with appropriate attributes. When registered as an owner, SK with attribute $Owner_i$ is issued; when registered as a user, SK with attributes such as *Physician* or *Researcher* is issued according to the application content.

We employ Ciphertext-Policy ABE (CP-ABE), in which decryption conditions are bound to the ciphertext, as it enables verifying that user attributes satisfy the policy associated with the data.

4) *IPFS as a Data Storage*: The proposed method uses IPFS (InterPlanetary File System) as a distributed data store, which improves availability and fault tolerance. IPFS uses content-based addressing through Content Identifiers (CIDs), enabling automatic tampering detection.

B. Data Encryption and Decryption Flow

By using both NFTs and ABE, the proposed method achieves higher security. The overall encryption/decryption flow is shown in Fig. 3. Note that both $NftKey$ and $AbeKey$ are symmetric keys.

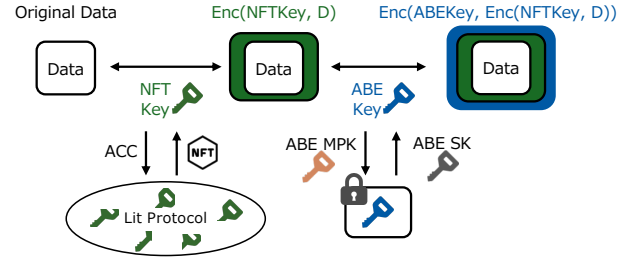


Fig. 3: Data Encryption and Decryption Flow

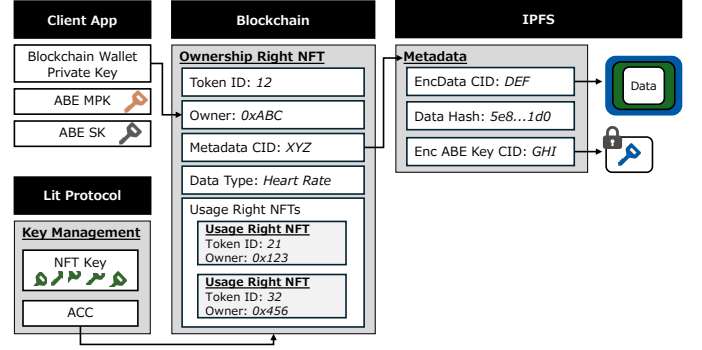


Fig. 4: Data Structure and Location

First, the data is encrypted with $NftKey$. $NftKey$ is managed in a distributed manner by Lit Protocol, with an ACC specifying the condition “the user holds either the ownership NFT or the usage NFT corresponding to the target data.” Next, the resulting ciphertext is additionally protected with ABE. Because ABE is computationally expensive, the data itself is encrypted with $AbeKey$, and then $AbeKey$ is ABE-encrypted as $AbeEnc(MPK, AbeKey, Attr)$ to obtain $EncAbeKey$.

For decryption, the user first derives $AbeKey$ by computing $AbeDec(SK, EncAbeKey)$. Then, the user queries Lit Protocol to obtain $NftKey$. Using the two recovered keys, the original data is finally decrypted.

C. Data Structure and Location

Figure 4 illustrates the structure and placement of data in the proposed method; arrows indicate references. A notable point is the structure and placement of NFT-related data. Because on-chain storage is costly, data other than what must reside on-chain is placed off-chain. Accordingly, such data is extracted as metadata and stored on IPFS, and the resulting CID is recorded in the NFT. A Usage NFT is managed as a child element of the corresponding Ownership NFT, which clarifies the data associated with each Usage NFT.

D. System Workflow

Using the components described above, we explain how data registration, access granting, and data access are realized. In the algorithms, $Enc(Key, Data)$ and $Dec(key, EncData)$ denote encryption and decryption, respectively. Methods beginning with $\blacklozenge$ are executed on the blockchain. When the algorithms are executed, it is assumed that a signature generated by the caller's blockchain wallet is automatically attached; in

Algorithm 1 Data Registration

```
1: procedure MAIN(Data, DataType, Attr)
2:   CallerAddr  $\leftarrow$  msg.sender()
   #STEP1: NFT Minting
3:   TokenId  $\leftarrow$ 
      $\blacklozenge$ MintOwnerNFT(CallerAddr, DataType)
   #STEP2: NFT Encryption
4:   NftKey  $\leftarrow$  GetSymmetricKey()
5:   NftEncData  $\leftarrow$  Enc(NftKey, Data)
6:   RegisterLitKeyToLitProtocol(NftKey, ACC)
   #STEP3: ABE Encryption
7:   AbeKey  $\leftarrow$  GetSymmetricKey()
8:   AbeNftEncData  $\leftarrow$  Enc(AbeKey, NftEncData)
9:   EncAbeKey  $\leftarrow$  AbeEnc(MPK, AbeKey, Attr)
   #STEP4: Data Upload
10:  EncDataCid  $\leftarrow$  UploadToIPFS(AbeNftEncData)
   #STEP5: ABE Enc Key Upload
11:  EncAbeKeyCid  $\leftarrow$  UploadToIPFS(EncAbeKey)
   #STEP6: Metadata Creation and Upload
12:  DataHash  $\leftarrow$  Hash(Data)
13:  Metadata  $\leftarrow$  CreateMetadata(EncDataCid,
    DataHash, EncAbeKeyCid, Attr)
14:  MetadataCid  $\leftarrow$  UploadToIPFS(Metadata)
   #STEP7: SetTokenMetadata
15:   $\blacklozenge$ SetTokenMetadata(TokenId, MetadataCid)
16: end procedure
```

Algorithm 2 Access Granting by Owner's Consent

```
1: procedure  $\blacklozenge$ MAIN(TokenId, UserAddr)
2:   CallerAddr  $\leftarrow$  msg.sender()
3:   if CallerAddr is NOT the owner of TokenId then
4:     Abort ()
5:   end if
6:   UsageNftId  $\leftarrow$ 
     MintUsageNFT(TokenId, UserAddr)
7: end procedure
```

this context, msg.sender() returns the caller's wallet address derived from the signature.

1) *Data Registration*: The data registration flow is shown in Algorithm 1. It takes data *Data*, data type *DataType*, and allowed attribute set *Attr* as inputs. Let the executor be user *i*. By default, *Attr* is set to ($Owner_i \vee Physician$), and if the data is also intended to be used for research and development, $\vee Researcher$ is additionally included.

First, an ownership NFT is issued, and its *TokenId* is obtained. Next, as described in Section IV-B, the data is encrypted using *NftKey* and *AbeKey*. At this time, the ACC in Lit Protocol is configured to require "holding the NFT corresponding to *TokenId*." The encrypted data and the ABE-encrypted *AbeKey* are uploaded to IPFS. Metadata is then constructed from the CIDs of these data items and uploaded to IPFS. Finally, the metadata CID is set in the ownership NFT.

Algorithm 3 Access Granting Without Owner's Consent

```
1: procedure  $\blacklozenge$ MAIN(TokenId)
2:   CallerAddr  $\leftarrow$  msg.sender()
3:   DataType  $\leftarrow$  GetDataType(TokenId)
4:   if DataType is NOT in the pre-defined list then
5:     Abort ()
6:   end if
7:   UsageNftId  $\leftarrow$ 
     MintUsageNFT(TokenId, CallerAddr)
8: end procedure
```

Algorithm 4 Data Access

```
1: procedure MAIN(TokenId)
2:   CallerAddr  $\leftarrow$  msg.sender()
   #STEP1: Metadata Retrieval
3:   MetadataCid  $\leftarrow$  GetMetadataCid(TokenId)
4:   Metadata  $\leftarrow$  FetchFromIPFS(MetadataCid)
   #STEP2: Download Encrypted Data
5:   EncDataCid  $\leftarrow$  GetEncDataCid(Metadata)
6:   AbeLitEncData  $\leftarrow$  FetchFromIPFS(EncDataCid)
   #STEP3: Download ABE Enc Key
7:   EncAbeKeyCid  $\leftarrow$  GetAbeEncKeyCid(Metadata)
8:   EncAbeKey  $\leftarrow$  FetchFromIPFS(EncAbeKeyCid)
   #STEP4: ABE Decryption
9:   AbeKey  $\leftarrow$  ABE-Dec(sk, EncAbeKey)
10:  LitEncData  $\leftarrow$  Dec(AbeKey, AbeLitEncData)
   #STEP5: NFT Decryption
11:  NftKey  $\leftarrow$ 
     GetNftKeyFromLitProtocol(CallerAddr, TokenId)
12:  Data  $\leftarrow$  Dec(NftKey, LitEncData)
13: end procedure
```

2) *Access Granting*: The access-granting flow based on the owner's consent is shown in Algorithm 2. It takes as inputs the ownership NFT's *TokenId* and the wallet address *UserAddr* to which the usage-right NFT will be issued. After verifying that the caller is the owner of *TokenId*, a usage-right NFT is issued. In this process, the usage-right NFT is managed as a child element of the ownership NFT corresponding to *TokenId*.

Next, Algorithm 3 shows the flow for granting access rights to limited data categories without the owner's consent. A usage-right NFT is issued to the caller only if the data type *DataType* corresponding to the ownership NFT's *TokenId* is included in a predefined list of allowed data types.

All access-granting flows are executed by smart contracts on the blockchain. This ensures the correctness of the flow and prevents unauthorized access-right issuance.

3) *Data Access*: The data access flow is shown in Algorithm 4. It takes as input the *TokenId* of the ownership NFT corresponding to the target data. Note that a holder of a usage-right NFT can obtain the *TokenId* by referencing its parent ownership NFT. First, the metadata stored on IPFS is retrieved using the metadata CID recorded in the ownership NFT. Next, using the CIDs contained in the metadata, the

encrypted data and the ABE-encrypted *AbeKey* are obtained from IPFS. Finally, the original data is recovered by decrypting according to the procedure described in Section IV-B.

V. SECURITY VERIFICATION

We verify the security of the proposed method by analyzing its behavior against representative attack scenarios based on the algorithms. The following assumptions are made: Lit Protocol always behaves honestly. The ABE server issues *SK* with appropriate attributes corresponding to each user. Data decryption is impossible without the legitimate keys.

a) *Unauthorized Access without NFTs*: If an NFT is not held, the user cannot obtain *NftKey* from Lit Protocol in STEP 5 of Algorithm 4, and thus cannot decrypt the data.

b) *Unauthorized Access with NFTs*: Even if an NFT is obtained illegitimately, the user cannot decrypt the data without *SK* that has the appropriate attributes, because *AbeKey* cannot be derived in STEP 4 of Algorithm 4.

c) *Unauthorized Access Granting*: Algorithm 2 can only be executed by the owner of the ownership NFT, and therefore a usage-right NFT cannot be issued to an unauthorized user. Algorithm 3 can be executed by anyone, but access rights can only be granted for predefined data categories. In addition, the issuance history of usage-right NFTs is immutably recorded on the blockchain, enabling the owner to later verify which users accessed which data.

VI. IMPLEMENTATION AND EVALUATION

We implemented the proposed method and measured the processing time of each flow to conduct a performance evaluation.

A. Implementation Details

We implemented a prototype without a UI primarily in TypeScript. Since no TypeScript library was available for ABE, we implemented the ABE component in Python using available cryptographic libraries. For the ABE curve, we used SS512, which is also adopted in other studies [13]. For *Attr*, we set $Owner_i \vee Physician$. Communication between TypeScript and Python was performed via a local HTTP server. We used the datil-testnet of Lit Protocol. AES-256-GCM was used as the symmetric key encryption algorithm. Pinata was used as the IPFS access gateway. For the blockchain, we used the Ethereum Sepolia testnet. As described later, Sepolia exhibited low performance; therefore, we also used the Base Testnet to evaluate performance differences. The NFTs were implemented in Solidity and deployed on each blockchain.

B. Evaluation Setup and Method

The evaluation machine was equipped with an Apple M4 Max 14-core CPU and 36 GB of memory, and connected to a 600 Mbps network. We prepared three data sizes: 10 KB, 1 MB, and 100 MB. For each data size, we measured the time for data registration, access-granting, and data access. Since we confirmed that access-granting with and without explicit consent exhibited identical processing time, we only show

TABLE I: Processing Time for Each Operation on Sepolia

	10KB	1MB	100MB
Data Registration	23.16s	23.88s	48.16s
Access Granting		13.87s	
Data Access	0.99s	1.07s	14.07s

the results for access-granting without explicit consent. Each experiment was performed five times. However, because many core components are decentralized systems and exhibited large runtime variation across trials, we discarded the fastest and slowest results and used the average of the remaining three measurements.

For reference, when retrieving one month of sleep logs using the Web API of Fitbit, a representative wearable device, the measured data size was 99.79 KB. Therefore, we consider the processing time for the 1 MB dataset as a practical reference value.

C. Result

Table I shows the processing time required for each operation. Since access granting is independent of data size, we present only the result for the 1 MB case.

Data registration takes 23.88 seconds for 1 MB, which appears too slow for practical use. Therefore, we measured the time required for each processing step, and the results are shown in Fig. 5. The results show that the blockchain-dependent processes in STEP 1 and STEP 7 of Algorithm 1 each take approximately 10 seconds regardless of data size. It is well known that blockchain latency is large, and these results are consistent with prior observations [14]. Note that the Sepolia testnet used in this experiment has an average transaction interval of approximately 12 seconds. To evaluate the impact of transaction latency, we conducted the same experiment on the Base Testnet, which has a shorter transaction interval of approximately 2 seconds. The results are shown in Fig. 6. By significantly reducing blockchain-dependent processing time, the total registration time for 1 MB was reduced to 12.08 seconds. This indicates that using a faster blockchain can substantially reduce data registration time. Furthermore, if STEP 1 and STEP 7 of Algorithm 1 are consolidated to reduce blockchain transactions from two to one, further acceleration is expected. Note that for a data size of 100 MB, NFT encryption in Step 2 and data upload in Step 4 become bottlenecks, so improving blockchain performance alone may not be sufficient to achieve a significant reduction.

Access granting is also affected by the transaction interval on Sepolia, taking 13.87 seconds. On the Base Testnet, this was reduced to 6.01 seconds. Thus, access granting time can also be shortened by using a faster blockchain.

Data access requires approximately one second for data up to 1 MB, which is acceptable for practical use. For 100 MB, approximately 14 seconds are required, of which 12.38 seconds occur in Step 5 of Algorithm 4. Thus, the operations related to Lit Protocol are likely the primary bottleneck in this case.

In summary, for the 1 MB data size, which serves as a practical reference, data access takes approximately one second,

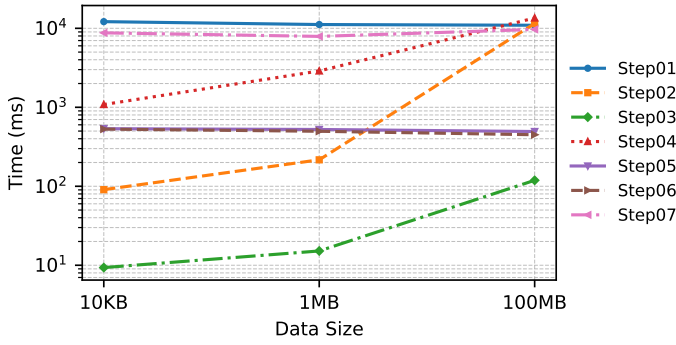


Fig. 5: Data Registration Time vs. Data Size on Sepolia

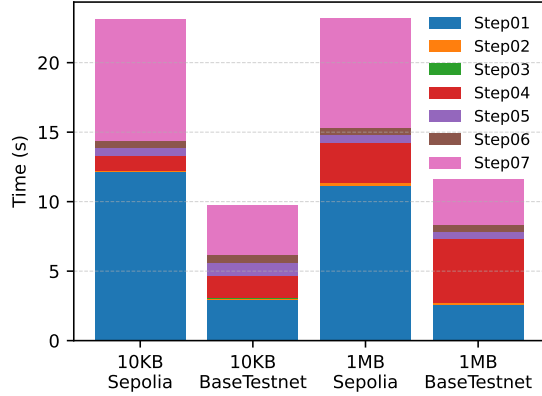


Fig. 6: Data Registration Time Comparison with Sepolia vs BaseTestnet

indicating that it is practically acceptable from a processing-time perspective. Data registration and access granting take approximately 12 seconds and 6 seconds, respectively, on the Base Testnet. Additionally, by reducing the number of blockchain operations or using a faster blockchain, we confirmed that these processes can be further accelerated, making processing times even more practical.

VII. CONCLUSION AND FUTURE WORK

In this paper, we proposed a method that strengthens security through hybrid encryption using NFTs and ABE, while enabling data retrieval without explicit user consent for limited data categories. We verified the security of the proposed method against representative attack scenarios, demonstrating that secure data access and rights management are ensured. We also implemented the proposed method and measured the processing time of each flow for performance evaluation. For the 1 MB data size, which serves as a practical reference value, data access was completed in approximately one second, showing that it is within a practically acceptable range. On the other hand, data registration and access granting took approximately 12 seconds and 6 seconds, respectively, on the Base Testnet. Through detailed analysis of processing time, we confirmed that these operations can be accelerated by reducing the number of blockchain transactions or using a faster blockchain, and thus can achieve practical processing times.

As future work, we plan to develop a PHR-sharing application based on the proposed method and conduct usability evaluation with real users. In addition, since each experimental trial in the performance evaluation was conducted independently, it is necessary to analyze the performance under concentrated workloads, such as during disaster scenarios, to verify applicability in high-load conditions. Optimizing each component of the proposed method for larger data sizes is also an important future challenge.

ACKNOWLEDGMENT

These research results were obtained from the commissioned research(No.23801) by National Institute of Information and Communications Technology (NICT), Japan.

REFERENCES

- [1] D. Sarwal and V. Gupta, "Personal health record," in *StatPearls [Internet]*. Treasure Island, FL: StatPearls Publishing, Jan 2025, updated 2024 Sep 10. [Online]. Available: <https://www.ncbi.nlm.nih.gov/books/NBK557757/>
- [2] A. Hosseini, H. Emami, Y. Sadat, and S. Paydar, "Integrated personal health record (phr) security: requirements and mechanisms," *BMC Medical Informatics and Decision Making*, vol. 23, no. 116, 2023.
- [3] M. A. Saberi, H. McHeick, and M. Adda, "From data silos to health records without borders: A systematic survey on patient-centered data interoperability," *Information*, vol. 16, no. 2, 2025. [Online]. Available: <https://www.mdpi.com/2078-2489/16/2/106>
- [4] N. Miyanishi, M. Tani, R. Nishitsuji, Y. Shijo, and K. Yamada, "Health information management through blockchain and nfts: Experimental evaluation of data sharing reliability," in *2025 17th Biomedical Engineering International Conference (BMEiCON)*, 2025, pp. 1–5.
- [5] M. Tani, N. Miyanishi, R. Nishitsuji, Y. Shijo, and K. Yamada, "Health information management through blockchain and nfts: Experimental evaluation of data sharing performance," in *2025 17th Biomedical Engineering International Conference (BMEiCON)*, 2025, pp. 1–5.
- [6] J. Kaisto, T. Juutilainen, and J. Kauranen, "Non-fungible tokens, tokenization, and ownership," *Computer Law & Security Review*, vol. 54, p. 105996, 2024. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0267364924000633>
- [7] R. Conforti, "Informatics in emergency medicine: A literature review," *Emergency Care and Medicine*, vol. 2, no. 1, 2025. [Online]. Available: <https://www.mdpi.com/2813-7914/2/1/2>
- [8] M. George and A. M. Chacko, "Health Passport: A blockchain-based PHR-integrated self-sovereign identity system," *Frontiers in Blockchain*, vol. 6, 2023. [Online]. Available: <https://doi.org/10.3389/fbloc.2023.1075083>
- [9] T. M. Kim, T. Ko, B. W. Hwang, H. G. Paek, and W. Y. Lee, "Self-sovereign management scheme of personal health record with personal data store and decentralized identifier," *Computational and Structural Biotechnology Journal*, vol. 28, pp. 16–28, 2025. [Online]. Available: <https://doi.org/10.1016/j.csbj.2024.11.036>
- [10] T. Klementi, G. Piho, and P. Ross, "A reference architecture for personal health data spaces using decentralized content-addressable storage networks," *Frontiers in Medicine*, vol. 11, 2024. [Online]. Available: <https://doi.org/10.3389/fmed.2024.1411013>
- [11] B. M. Yakubu, S. M. Ali, M. I. Khan, and P. Bhattarakosol, "Patcen: A blockchain-based patient-centric mechanism for the granular access control of infectious disease-related test records," *PLOS ONE*, vol. 19, no. 9, pp. 1–35, 09 2024. [Online]. Available: <https://doi.org/10.1371/journal.pone.0310407>
- [12] "Lit protocol," <https://www.litprotocol.com/>, accessed: 2025-10-28.
- [13] A. Singh and G. Rathee, "Fate: flexible attribute-based traceable encrypted data sharing scheme using smart contracts in wireless medical sensor networks," *Annals of Telecommunications*, vol. 80, pp. 323–339, 2025.
- [14] M. Shaikh, S. A. Memon, A. Ebrahimi, and U. K. Wiil, "A systematic literature review for blockchain-based healthcare implementations," *Healthcare*, vol. 13, no. 9, 2025. [Online]. Available: <https://www.mdpi.com/2227-9032/13/9/1087>