sensors

MDPI

Article

Blockchain-Based Access Control Scheme for Secure Shared Personal Health Records over Decentralised Storage

Hassan Mansur Hussien [1,*], Sharifah Md Yasin [1,2,*], Nur Izura Udzir [1] and Mohd Izuan Hafez Ninggal [1]

[1] Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang 43400, Malaysia; izura@upm.edu.my (N.I.U.); mohdizuan@upm.edu.my (M.I.H.N.)
[2] Institute for Mathematical Research (INSPEM), Universiti Putra Malaysia, Serdang 43400, Malaysia
* Correspondence: hassanalobady@gmail.com (H.M.H.); ifah@upm.edu.my (S.M.Y.); Tel.: +60-182015604 (H.M.H.)

Abstract: Blockchain technology provides a tremendous opportunity to transform current personal health record (PHR) systems into a decentralised network infrastructure. However, such technology possesses some drawbacks, such as issues in privacy and storage capacity. Given its transparency and decentralised features, medical data are visible to everyone on the network and are inappropriate for certain medical applications. By contrast, storing vast medical data, such as patient medical history, laboratory tests, X-rays, and MRIs, significantly affect the repository storage of blockchain. This study bridges the gap between PHRs and blockchain technology by offloading the vast medical data into the InterPlanetary File System (IPFS) storage and establishing an enforced cryptographic authorisation and access control scheme for outsourced encrypted medical data. The access control scheme is constructed on the basis of the new lightweight cryptographic concept named smart contract-based attribute-based searchable encryption (SC-ABSE). This newly cryptographic primitive is developed by extending ciphertext-policy attribute-based encryption (CP-ABE) and searchable symmetric encryption (SSE) and by leveraging the technology of smart contracts to achieve the following: (1) efficient and secure fine-grained access control of outsourced encrypted data, (2) confidentiality of data by eliminating trusted private key generators, and (3) multikeyword searchable mechanism. Based on decisional bilinear Diffie–Hellman hardness assumptions (DBDH) and discrete logarithm (DL) problems, the rigorous security indistinguishability analysis indicates that SC-ABSE is secure against the chosen-keyword attack (CKA) and keyword secrecy (KS) in the standard model. In addition, user collusion attacks are prevented, and the tamper-proof resistance of data is ensured. Furthermore, security validation is verified by simulating a formal verification scenario using Automated Validation of Internet Security Protocols and Applications (AVISPA), thereby unveiling that SC-ABSE is resistant to man-in-the-middle (MIM) and replay attacks. The experimental analysis utilised real-world datasets to demonstrate the efficiency and utility of SC-ABSE in terms of computation overhead, storage cost and communication overhead. The proposed scheme is also designed and developed to evaluate throughput and latency transactions using a standard benchmark tool known as Caliper. Lastly, simulation results show that SC-ABSE has high throughput and low latency, with an ultimate increase in network life compared with traditional healthcare systems.

Keywords: blockchain; decentralised storage; data privacy; attribute-based encryption; searchable encryption; access control; chosen-keyword attack; standard adversary model

check for updates

Citation: Hussien, H.M.; Yasin, S.M.; Udzir, N.I.; Ninggal, M.I.H. Blockchain-Based Access Control Scheme for Secure Shared Personal Health Records over Decentralised Storage. *Sensors* **2021**, *21*, 2462. https://doi.org/10.3390/s21072462

Academic Editor: Valderi R. Q. Leithardt

Received: 2 January 2021
Accepted: 15 February 2021
Published: 2 April 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Blockchain technology has gained considerable attention in many industrial and academic aspects. In particular, the merging of blockchain technology and smart contracts has enabled a ubiquitous decentralised interaction of nodes, thereby yielding an applaudable opportunity for certain applications in private and public domains, such as healthcare systems [1], biomedical sciences [2], and smart cities [3]. In healthcare systems, blockchain

technology has played a crucial role in transforming network infrastructure into a stable, secure, auditable, and decentralised environment. The sharing of personal health records (PHRs) is vital for diagnosis and disease care to facilitate patients' treatment by various medical professionals. PHR systems have become the standard technology that handles the proliferation of generated medical records whilst maintaining the required quality of services. Furthermore, potential blockchain technology enablers, such as decentralised networks, transactions, consensus mechanisms, and smart contracts, can improve security and integrity. However, issues that concern blockchain or smart contracts are related to privacy and storage capacity [4–7]. The development of PHR applications on the blockchain network may enable anyone to access transaction data due to the blockchain's transparent feature. This feature has raised privacy concerns regarding the Health Insurance Portability and Accountability Act HIPAA requirements and the ability of patients to participate in the publication of their personal information in the blockchain network. By contrast, blockchain requires considerable storage to record whole transactions in the network; such requirement can be a problem for restrictive nodes that send data to the network. Blockchain can ensure that the stored and shared PHRs are not manipulated, unforgeable and verifiable but can effectively suffer from storage requirements of large-scale distributed data [8–10].

An important detail to consider is to examine the advantages and disadvantages of using blockchain technology for PHR against a range of perspectives, such as security, privacy and storage capacity. Recent findings tend to resolve the above issues by storing medical databases on offline storage, such as cloud servers, and by setting up an access control scheme to prevent unauthorised users from manipulating data through leveraging the attribute-based cryptosystem [11–13] reproxy encryption [14,15] and smart contracts [16–18] to control the users' privilege. At the same time, other researchers have attempted to offload the actual large-scale distributed data into the InterPlanetary File System (IPFS) storage without setting up any enforced cryptographic access control [19,20]. Meanwhile, concepts of outsourced blockchain databases on honest-but-curious third-party storage, such as cloud servers, are considered a double-edged sword technique that resolves the scalability issue. However, it decreases the security level of blockchain against fully decentralised infrastructure and increases the level of single security failure. Simultaneously, the outsourcing of databases on IPFS decentralised storage eliminates the unreliable storage of third parties. Nevertheless, IPFS has a noticeable security flow that anyone with the hash of the file stored therein can easily retrieve it due to IPFS native workflow. In conclusion, the health data generated by patients are not well suitable for being stored in IPFS unless data are encrypted individually prior to outsourcing to the IPFS. Therefore, providing security and privacy to PHR systems with fine-grained access control is essential to support a technique that searches for encrypted data on the IPFS storage.

Cryptographic primitive methods are considered to be the appropriate solution for confidentiality of outsourced data. However, traditional cryptographic primitive methods, such as symmetric-key encryption and public-key encryption, cannot maintain effective access control over outsourced encrypted data. The ciphertext-policy attribute-based searchable encryption (CP-ABSE) has considerable advantages over other searchable encryption schemes in terms of the construction of secure fine-grained access control for outsourced encrypted data. CP-ABSE is a suitable scheme for storing medical data in the IPFS node by enabling fine-grained access control in an encrypted electronic format to control user privilege and support one-to-many scenarios. This scheme can support expressive access policies by determining any access structures. This scheme also provides a high level of data flexibility because the secret keys (SKs) of the data consumers can be generated at once and can be used to decrypt all the reverent ciphertext. On the contrary, traditional encryption schemes require a trusted private key generator (PKG) to initialise and distribute an SK to users. The PKG may cause serious issues with user data ownership, such as key misuse, data leakage, and capability to control their own data, due to its ability to decrypt all outsourced data stored on the server. Meanwhile, the current scheme has

suffered from expensive computational operations in its data outsourcing and retrieval aspects. At the same time, a secure and efficient conjunctive keyword search mechanism for CP-ABSE is essential for the exchange of real data to ensure that only authorised entities can access medical data. In addition, most of the current schemes are not considered the adversary model of security resistance against the chosen keyword attack (CKA) and keyword secrecy (KS) in the standard model.

1.1. Motivation

The emergence of PHR sharing systems using cloud technology can provide patients with a complete and accurate online personal medical history, which can benefit patients, research institutions, pharmaceutical companies, and the entire healthcare system. Under these circumstances, the patient's PHRs are often outsourced to the third party, such as the cloud service provider, to achieve resource sharing and reduce the data centre's maintenance costs. This situation leads to security issues on how to ensure the security, privacy and searchability of PHRs. To overcome this problem, some researchers are attempting to combine searchable symmetric encryption and ciphertext-policy attribute-based encryption. However, this hybrid encryption scheme requires centralised key management in a cloud server, leading to a single point of security failure because the cloud platform may not be credible due to employee corruption or a threat to the authorisation centre. Fortunately, the use of blockchain technology and smart contracts can easily and securely manage key management and distribution. The concept of creating a permanent and decentralised way to store and share files on IPFS can be perfectly aligned with the blockchain for the provision of a decentralisation infrastructure. Additional blockchain features, such as unforgeable and tamper-proofing of stored data, are also an advantage. Therefore, a secure and efficient CP-ABSE must be designed to support a multikeyword search fine-grained access control for PHRs in the blockchain over IPFS storage without relying on a centralised authority by ensuring its resistance to CKAs and keyword secrecy underneath the standard model.

1.2. Contributions

To overcome the aforementioned challenge, this study proposes a new lightweight cryptographic concept called smart contract-based attribute-based searchable encryption (SC-ABSE) by combining ciphertext-policy attribute-based encryption (CP-ABE), searchable symmetric encryption (SSE), smart contract, and IPFS storage. The proposed SC-ABSE eliminates the need for trusted PKGs from the system by allowing the data owner to distribute SKs to data users so that they can control their outsourced encrypted data stored in IPFS; such approach would be more effective than the traditional CP-ABSE schemes. At the same time, the smart contract in the blockchain is used to maintain the SK of users, and the problem of key management in the traditional CP-ABE schemes is resolved. SC-ABSE solves the current problems of developing PHR applications based on blockchain technology by designing a secure and efficient authorisation and access control mechanism that allows patients to store their medical records in a decentralised storage repository (i.e., IPFS) whilst preventing unauthorised users from disclosing medical data. The main contributions of this study are as follows:

- The proposed SC-ABSE scheme removes trusted PKGs by achieving high privacy protection of users' SKs to ensure the consistent confidentiality of outsourced encrypted data in the IPFS storage. Moreover, it supports a secure multikeyword searchable fine-grained access control by providing one-to-many encryption to prevent unauthorised users from disclosing medical data in decentralised IPFS storage.
- The lightweight key generation algorithm is proposed in the SC-ABSE scheme in comparison with other existing schemes due to a reduced number of pairing operations in which it is a more constant secret key (SK). However, the smart contract is used to auto-enforce the synchronisation of nodes. Any modification in user key distributions can be detected automatically.

- The lightweight outsourcing mechanism is proposed in the SC-ABSE scheme by enabling the blockchain node's user–patient to encrypt their medical data using the advanced encryption standard (AES) and then only encrypt the symmetric key file and the keyword using attribute-based encryption. This approach reduces the computational complexity of the encryption algorithm by turning into constant ciphertext.
- The lightweight retrieving mechanism is proposed in the SC-ABSE scheme by designing a more secure and efficient token generation algorithm in comparison to other existing schemes. This is due to shifting almost all of the computational complexity processes of the pairing operations to the search part of the IPFS storage entity. Generally, medical data decryption does not depend on the number of attributes in the access control policies.
- The proposed scheme is proven to be secure against CKAs and keyword secrecy under the hardness assumptions of DBDH and DL problems, respectively, in the standard model. Moreover, the formal security verification method using the Automated Validation of Internet Security Protocols and Applications (AVISPA) tool verifies that the reinforced security validation of the proposed scheme withstands replay and MIM attacks.
- A series of comprehensive experimental investigations are performed in terms of computational costs, storage costs, communication costs, throughput, and latency transactions, demonstrating that the proposed scheme achieves a higher level of security with less computational complexity costs than other existing state-of-the-art schemes.

1.3. Organisation

The remaining sections are organised as follows: The related work is presented in Section 2. The technical preliminaries of the study are described in Section 3. The proposed scheme and its security definition are presented in Section 4. The security analysis of the proposed scheme is demonstrated in Section 5. The performance analysis of the proposed scheme given in Section 6. Lastly, the conclusion and open directions are summarised in Section 7.

2. Related Work

2.1. Blockchain-Based Searchable Encryption

The searchable symmetric encryption scheme was first proposed by [21] to ensure that the storage server cannot learn any piece of sensitive information by involving a secure and efficient query of keywords associated with encrypted information. The study [22] proposed a new security definition for searchable symmetric encryption without considering the security of token generation and search queries against keyword secrecy. In [23,24], a new searchable symmetric encryption was presented by designing a secure and efficient scheme based on the order-preserving encryption technique. Meanwhile, [25–29] considered a multikeyword retrieval scheme using the AND gate to access various keywords and protect user information and search tokens, thus maintaining security. The major aim of these approaches is to improve the security of SSE schemes in terms of promoting and expanding their capabilities in a multiuser environment. In conclusion, the literature acknowledges that current SSE schemes are unavoidable in terms of security of CKAs and keyword secrecy due to the need for a strict access policy for outsourced encrypted pieces of information. However, the SSE scheme suffers from critical and expensive key distribution problems. Meanwhile, the SSE scheme enables clients to generate a searchable ciphertext stored on a third-party server, and the server is responsible for searching the keywords associated with the ciphertext. Nevertheless, these schemes require a strict access policy to control the search for outsourced encrypted data on the server due to data owners being unable to grant search rights to their data.

Recent interest in searchable symmetric encryption has given a new impetus to the use of blockchain for improving security on cloud computing and health system

research [30–32]. These studies have constructed a searchable symmetric encryption scheme by leveraging blockchain features, such as transactions, to retrieve the data stored in cloud servers. The proposed scheme allows users to outsource encrypted data securely to honest-but-curious cloud storage and enables data consumers to search and retrieve data. Moreover, [33] introduced a novel Searchain concept based on the blockchain and keyword search system by enabling users to search securely over a set of keywords attached to data stored in a decentralised repository. In addition, [34–36] revealed its tremendous benefits from blockchain technology to develop a secure, searchable encryption system to fulfil the security matter of sharing medical data. Moreover, current SSE schemes require crucial and costly key distribution and cannot support one-to-many encryption. As a result, these shortcomings of the SSE cryptographic primitives have become an improper algorithm for being completely implemented as an independent security mechanism for healthcare applications in blockchain technology.

2.2. Blockchain-Based Ciphertext Policy Attribute-Based Encryption

The ideology of attribute-based encryption (ABE) has gained considerable attention from scholars since it was released in 2006 [37] due to its marvelous characteristics that allow robust fine-grain access control over outsourced encrypted data depending on user attributes. CP-ABE is an ABE variant that allows users to develop an access policy strictly for their data by determining the set of attributes [38]. Numerous researchers have contributed towards CP-ABE to strengthen its capabilities in terms of security [39–41] and efficiency [42–44] and expand its functionality of supporting searchable mechanisms [45–47] to become suitable cryptographic primitives for certain system requirements. These schemes tend mainly to improve communication efficiency and reduce the cost of computational user complexity to access the encrypted data stored in the cloud by utilising the outsourced computation and storage server-aided technique. In addition, user revocation and attribute revocation updates have been achieved, considering the length of ciphertext that could result in high communication costs in practical applications. Simultaneously, the security resistance of these schemes to ciphertext or plaintext chosen attacks is maintained. Lately, several researchers have begun to explore the technology of CP-ABE schemes with medical records to ensure the data security of the exchange of medical data among multiple healthcare providers in a cloud-based environment by developing a secure and efficient system for accessing data [48–52]. The primary objective of these approaches is to obtain a feature of the access structure policy of the CP-ABE and ensure that medical data can only be accessed by matching user attributes. The era of blockchain technology research has moved towards adopting ABE to secure some of its drawbacks in terms of healthcare application [53,54]. These studies have utilised the CP-ABE scheme with blockchain to achieve user authorisation with high flexibility and efficiency for telemedicine systems by utilising the distributed independent key to update the patient keys with multiple healthcare providers to be matched in a real situation.

The conventional CP-ABSE approach presented in Table 1 still needs a trusted PKG. The SK generated by the PKG for users is not adequately flexible. It can lead to misuse of the key by compromising user privacy. To conduct search operations over outsourced encrypted data, these schemes rely on the cloud server. The cloud server can return inaccurate results or even no results to save resources. In comparison, the proposed SC-ABSE scheme uses the IPFS decentralised storage method to resolve a single point of security failure in conventional cloud storage and pressure IPFS to perform the search correctly. Furthermore, the trusted PKG is removed. Another challenge faced by conventional CP-ABSE is the queries of the outsourced encrypted data over the server. The single-keyword search is a trivial procedure in which each keyword is performed separately, resulting in inefficient queries and some information leak to the server. By contrast, multikeyword search allows a user to obtain encrypted data over the sever by attaching several keywords during one single query. Moreover, the proposed SC-ABSE scheme supports secure multikeyword searchable mechanisms without compromising security resistance to chosen-keyword

attack (CKA) and keyword secrecy (KS) in the standard model under the DBDH hardness assumption. By contrast, other schemes, [55–59] are analysed in the random oracle model, which is much weaker than the standard model. Such weakness is mainly due to the existence of a trusted PKG, which causes the key escrow problem by compromising the privacy of the user and leads to learning the data user's search information by testing all token generation and ciphertexts of keywords one by one. However, the CP-ABSE schemes proposed in [55–59] have a high computational complexity that burdens user experience in key generation, token generation, retrieval, and outsourcing of data over cloud storage environments by performing additional computational tasks. This high computational complexity can be a bottleneck for the users to upload and share medical data over an IPFS storage in a blockchain environment. A comparison of these approaches with the proposed scheme in terms of security properties and functionality features is presented in Table 1.

Table 1. Comparison of functional and security properties in various schemes.

Functionality Features	[55]	[56]	[57]	[58]	[59]	Current Study
Lightweight key generation	×	×	×	×	×	√
Lightweight outsourcing mechanism	√	×	×	×	×	√
Lightweight token generation	×	×	×	×	×	√
lightweight retrieving mechanism	×	×	×	×	×	√
Multiowner search	×	×	×	×	√	√
Multikeyword search	×	×	×	×	√	√
Fine-grained keyword search	×	×	×	×	×	√
Decentralised storage	×	×	√	×	×	√
Blockchain based	√	√	√	√	×	√
Eliminates trusted PKG	×	×	×	×	×	√
Security Properties						
Keyword secrecy (KS) in the standard model	×	×	×	×	×	√
Chosen-keyword attack(CKA) in the standard model	×	×	×	×	×	√
Man-in-the-middle attack in the AVISPA toolset	×	×	×	×	×	√
Replay attack in the AVISPA toolset	×	×	×	×	×	√
Tamper-proof resistance	×	×	×	×	×	√
User collusion attack resistance	×	×	×	×	×	√

3. Preliminaries

3.1. Bilinear Groups and Hardness Assumption Problems

For two multiplicative cyclic bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of a prime order p, a function $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is said to be a paired bilinear map if it satisfies the following properties:

- Bilinearity: For all $g \in \mathbb{G}_1$. and $a, b \in \mathbb{Z}_p$, we have $e\left(g^a, g^b\right) = e\left(g^b, g^a\right) = e(g,g)^{ab}$.
- Nondegeneracy: $g \in \mathbb{G}_1$. exists such that $e(g,g) \neq 1$.
- Computability: For any $g\ h \in \mathbb{G}_1$, an efficient polynomial-time algorithm must exist to compute the pairing $e(g,h)$.

Suppose $\mathcal{B}$ is a probabilistic polynomial-time (PPT) algorithm, where $\mathcal{B}(1^n) \rightarrow (n, p, \mathbb{G}_1\ \mathbb{G}_2\ , e)$ and $(p, \mathbb{G}_1\ \mathbb{G}_2\ , e)$ are considered an interchangeable variable, and n is a system security parameter. Then, this current research uses the hardness assumption problems for β as follows:

The DL problem in $\mathbb{G}_1$ is defined as follows: given the contrapositive function $\mathcal{B}(1^n) \rightarrow (n, p, \mathbb{G}_1\ \mathbb{G}_2\ , e)$ and two consistent elements of $g \in \mathbb{G}_1$ and $\in \mathbb{Z}_p$, the DL assumption states that for any PPT adversary A, a negligible contrapositive function exists such that

$$\Pr(\text{adversary } (n, p, g, g^x\ \mathbb{G}_1\ \mathbb{G}_2, e) = x\ ) \leq \text{negligible(n)}. \quad (1)$$

The above-mentioned probabilistic in Equation (1) is taken upon the random selection of $g \in \mathbb{G}_1$ and $\in \mathbb{Z}_p$. The randomness is used in algorithms $\mathcal{B}$ and A, and n is the security parameter of the system.

DBDH problem: Let $g \in \mathbb{G}_1$ and $\alpha,\ \beta,\ \gamma \in \mathbb{Z}_p$ be the four random elements and the contrapositive function of $(n, p,\ \mathbb{G}_1\ \mathbb{G}_2\ , e)$ be an output of $\mathcal{B}(1^n)$. The assumption problem of DBDH states that for each PPT adversary A, a negligible contrapositive function exists such that

$$\left| \begin{array}{c} \Pr\left(\text{adversary}\ \left(n, p, g, g^{\alpha}, g^{\beta}, g^{\gamma}, g^{\alpha\beta\gamma}, \mathbb{G}_1\ \mathbb{G}_2\ , e\right) = 1\right) - \\ \Pr(\ \text{adversary}\ \left(n, p, g, g^{\alpha}, g^{\beta}, g^{\gamma}, g^{z}, \mathbb{G}_1\ \mathbb{G}_2, e\right) = 1\ ) \leq\ \text{negligible}(n) \end{array} \right| \quad (2)$$

The above-mentioned probabilistic in Equation (2) is taken upon the random selection of $g \in \mathbb{G}_1$ and $\alpha,\ \beta,\ \gamma,\ z\ \mathbb{Z}_p$. The randomness is used in algorithms $\mathcal{B}$ and A, and n is the security parameter of the system.

3.2. Access Control

The access tree's inner nodes are threshold values, and each leaf node refers to a different set of the attribute. First, assume $\mathbb{A}$ is a tree access structure policy, and each nonleaf node is considered a threshold gate value for children's nodes. The number of children for node x is Ch_x, and the threshold value is K_x. However, if the value of $K_x = 1$, then the threshold is considered an OR gate; if the value of $K_x = Ch_x$, then the threshold is considered an AND gate that in a matter of the $0 <\ K_x \leq Ch_x$, whereby each leaf node be the representative of an attribute associated with its threshold value of $K_x = 1$. Afterwards, the parent of node x and the set of attributes associated with the leaf node are denoted as $parent(x)$ and attribute (x), respectively. The assigned number of $\{1, 2, \ldots, Ch_x\}$ is an index by node y, where the node y represents a child of node x. These index values are assigned uniquely in an arbitrary manner to the corresponding nodes in terms of the access control structure and are performed as follows:

- If the set of attributes is successfully satisfied with the tree access structure policy $\mathbb{A}$, then $\mathbb{A}_x\ (y) = 1$.
- If x is a nonleaf node recursively computed on $\mathbb{A}_{x'}(y)$ for the entire children x' of node x, then $\mathbb{A}_x\ (y)$ can return 1 if and only if as a minimum K_x children return 1.
- If x is a leaf node, then $\mathbb{A}_x\ (y)$ can return 1 if and only if $attributes(x)\ \in \gamma$.

3.3. Functional Encryption of the SC-ABSE Scheme

Assume MD is a message space of medical data, $\boldsymbol{g}$ is the access structure space, $\mathcal{U}^s$ is the space attributes set, S is the space of the attribute set and $(\ \mathbb{A})$ is the access structure policy. The SC-ABSE scheme consists of six algorithms as follows:

- Setup $\left(\mathbf{1^k}, \boldsymbol{\mu}\right) \rightarrow \boldsymbol{Pk}, \boldsymbol{MK}$**.** This algorithm takes a security parameter of 1^k and a set of universal attributes $\boldsymbol{\mu}$ as input. Then, it generates a public key PK and a master key MK.
- $KeyGeneration\ (\boldsymbol{S}, \boldsymbol{MK}\ ) \rightarrow \boldsymbol{SK}$. This algorithm takes a defined attribute set (S) and MK as input. This algorithm outputs the SK for users in the healthcare user node (HUN).
- $Encryption\ (\ \boldsymbol{PK}, \boldsymbol{MD}, \boldsymbol{kw},\ \mathbb{A}) \rightarrow \left(\text{MD}^*,\ SKE_{file},\ \text{I}\right)$. This algorithm takes the public key (PK) of users in the HUN, medical data of the patients (MD), the keyword (kw) and a specified access structure $(\mathbb{A})$ as input. This algorithm outputs the encrypted medical data (MD^*), the symmetric encryption key for medical data (SKE) and the encrypted indexed data (I). Then, it returns the tuple of $(\text{MD}^*,\ SEK,\ \text{I})$ to IPFS storage.
- $GenerationToken\ (SK, S, KW) \rightarrow T(T_{kw}S)$. This algorithm takes the final secret decrypting keys of users in the HUN (SK), a set of attributed (S) and given interesting keywords KW as input. Afterwards, the algorithm outputs corresponding token $T(T_{kw}S)$.

- *Search* $(\mathrm{I}, T_{kw}, S, \mathbb{A}) \rightarrow (\mathrm{MD}^{*}\mathrm{SKE})$. This algorithm takes the given index I with token (T_{kw}) and access structure policy $(\mathbb{A})$ as input. Then, it matches the indexed keywords for the corresponding encrypted medical data with received interested token whatever S (resp., T_{kw}) has satisfied with $(\mathbb{A})$ (resp., I). Then, it sends the encrypted ciphertext of the relevant tuple file results to the requester.
- *Decryption* $\left(\mathrm{MD}^{*}, SKE_{file}\right) \rightarrow (MD)$. This algorithm is takes the relevant search results of encrypted medical data (MD*) and a symmetric encryption key for medical data (SKE_{file}) as input. Then, the requester recovers the message (MD).

4. Proposed SC-ABSE Scheme

Here, the typical overall system model for data format and access control protocol is described. Then, the security definition and construction of SC-ABSE are given.

4.1. System Model

The proposed SC-ABSE scheme for secure IPFS medical data sharing consists of five entities. These entities are described in Table 2.

Table 2. Entities of smart contract-based attribute-based searchable encryption (SC-ABSE).

Entities	Description
Blockchain (BC)	Establishes and registers each node entity, such as the patient, the HUN and the IPFS on the network. To impose an agreement on each entity node, a smart contract is used for auditing purposes of all record requests and access activities.
Patient Node (PN)	Initialises system security parameters, for instance, public key (PK), master keys (MK), and final SK for the users in the HUN associated with defined attributes set (S). The patient node users choose an access structure to encrypted personal information and build the corresponding keyword index and upload the result of the ciphertext attached with an index to the IPFS storage.
Healthcare Users Node (HUN)	The users in the HUN require enough attributes to access the outsourcing ciphertext in IPFS by fulfilling the access structure policy.
IPFS Node	Stores the outsourced encrypted medical data of patients
Transactions pool	Contains the unconfirmed transactions for uploading and access medical data stored in the IPFS

The structure of the proposed SC-ABSE scheme is exhibited in Figure 1, where each step number is described as follows:

(1) The patient node (PN) initialises the system by executing the setup algorithm with the aid of the smart contract to generate public parameters, such as PK and MK whilst publishing PK and keeping MK secret.
(2) The PN generates the final SK associated with the attribute set for each user participating in the HUN by using a smart contract to execute the SK generation algorithm.
(3) The patient in the node encrypts his medical data using AES and then encrypts the symmetric key using ABE, generates the keyword for that encrypted data, encrypts it with the ABE algorithm and sends the tuple of the file, including encrypted medical data, the symmetric encryption key for medical data and the encrypted indexed keyword data to be stored in the outsourced IPFS.
(4) Upon receiving the tuple file from the patient, the IPFS stores the file into distributed node storage and returns the hash value for that file in the form of a transaction and adds it into the transaction pool for confirmation by the miners to append the transaction to the blockchain where each node will have that copy of the transaction. Meanwhile, hash values are stored in the form of a distributed hash table (DHT) smart contract.
(5) The users in the healthcare user's node search the patient ID via a transaction uploaded to the main blockchain network and generate a token by encrypting keywords of interest using the final SK received from the PN to obtain patient data and then send the attributes and token to the IPFS node.

(6) Upon receiving the token and attributes from the healthcare users, the IPFS searches the keywords if the attributes and token have been satisfied with an access policy and index keywords, respectively. Then, the IPFS sends the tuple file to the users.
(7) After receiving the tuple, the user in the HUN must derive the corresponding symmetric key to decrypt patient data.
(8) Unconfirmed transactions are uploaded for validation by the miners.
(9) Each transaction is signed and appended to the main blockchain network.

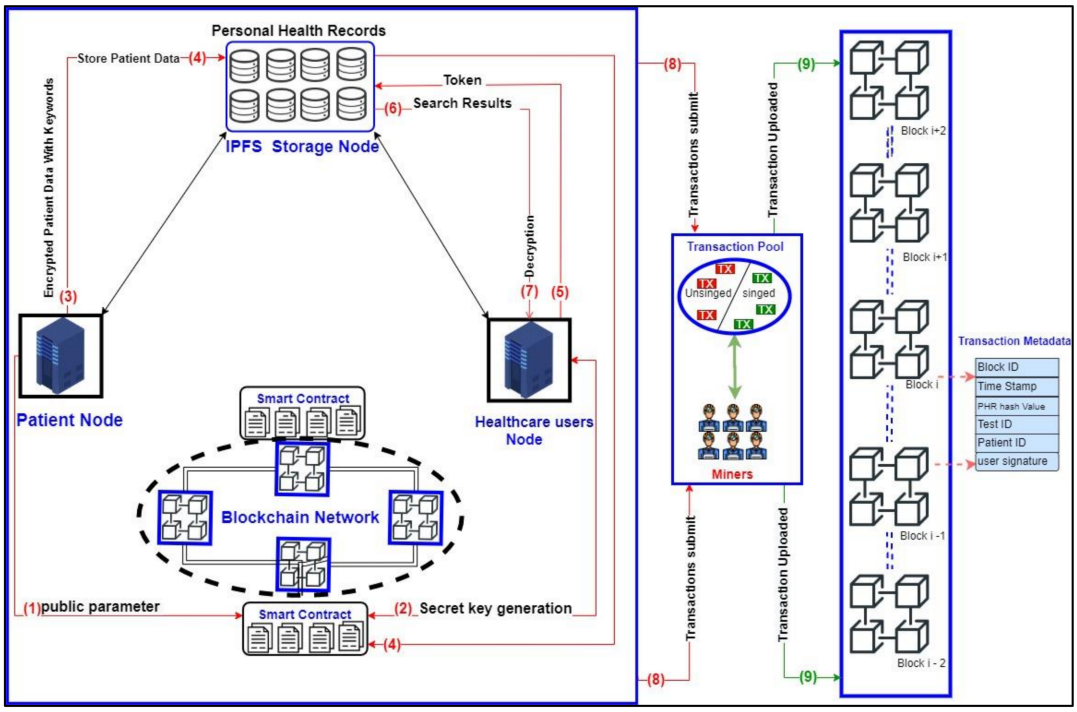


Figure 1. Proposed scheme architecture and system design.

4.2. Security Requirements and Threats

The security considerations of the proposed SC-ABSE scheme are described in this section.

1. CKA resistance: The proposed scheme can prove its security resistance by formalising the CKA's provable security indistinguishability. Assuming that the adversary is an IPFS storage, it can access any structure policy throughout the decryption process but cannot obtain information about the actual medical data stored there.
2. Keyword secrecy resistance: It ensures that malicious users in the HUN cannot compromise anything from the patient's medical data attached to the keyword or token search mechanism.
3. Tamper-proof resistance: The proposed scheme stores medical information, such as diagnostic results or patient history in the IPFS. The adversary intends perhaps to either alter some of the data or replace the existing with another data. User collusion attack resistance: SKs for users of the HUN are generated by the PN. Adversary users can generate a token by combining collusively with each other using their own SKs, attempting to retrieve patient data stored in the IPFS.
4. Replay and man-in-the-middle (MIM) attack resistance: Transactions are issued by the PN when the patient users upload data to the IPFS node. An adversary may copy an authorised user transaction from the blockchain or retrieve messages sent by an authorised user. Then, replay or MIM attacks occur when an adversary can change the communication messages on the PN to obtain the patient data stored in IPFS.

4.3. Smart Contracts

The smart contract is an entirely trusted component of the system and is a codable program that lives at the top of the blockchain. It can auto-enforce the code and perform the proposed SC-ABSE scheme managed by a P2P network of nodes. The transaction data structure for the contract is shown in Figure 2. Fundamentally, the access control mechanism is constructed on the basis of the proposed SC-ABSE cryptosystem, and it has four phases. In the system initialisation phase, public security parameters for the PHR system are generated. In the SK generation phase, a request for an algorithm of an SK is held to generate the final SKs for the users in the HUN. In the upload health data phase, the encrypted medical data are outsourced to the IPFS storage. In the access health data phase, a token and search and decryption process is constructed for the medical data stored in IPFS storage. The concrete construction of the above-mentioned phases has been described in detail in Section 4.5. The sequence diagram in Figure 3 specifies how the PN creates the contract. Then, the users in the PN, followed by users in the HUN, consents to the contract. The primary mechanism for the smart contract pattern is explained as follows:

- The PN creates the main contract to execute the SC-ABSE and generate public parameters (PK, MK and SK) for users in the HUN.
- The main contract requires the users in the PN to perform encrypted index keywords and store patient medical data into the IPFS.
- For users in the HUN side, when they wish to retrieve outsourced data in IPFS, the main contract can perform an aspect of the token generated, searched, and decrypted.

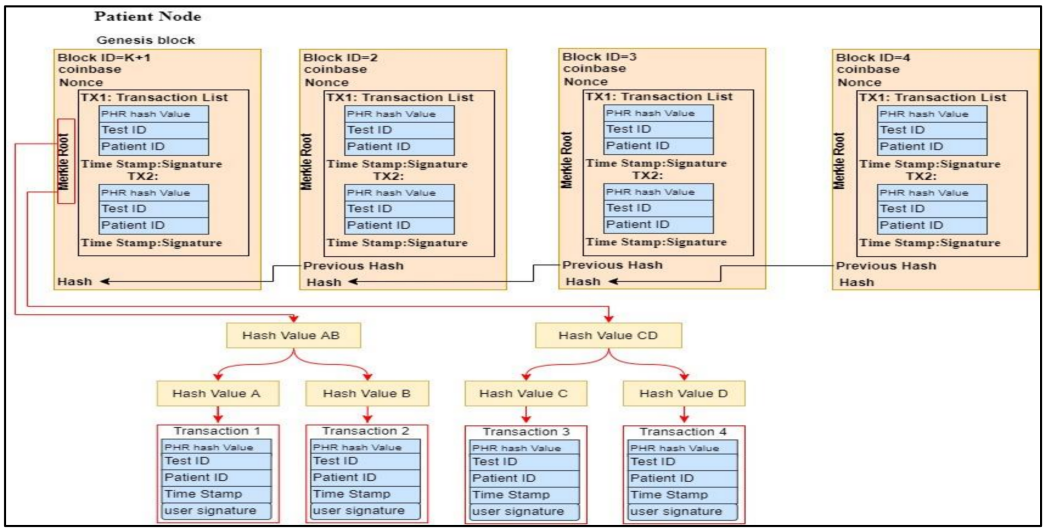


Figure 2. Data block structure.

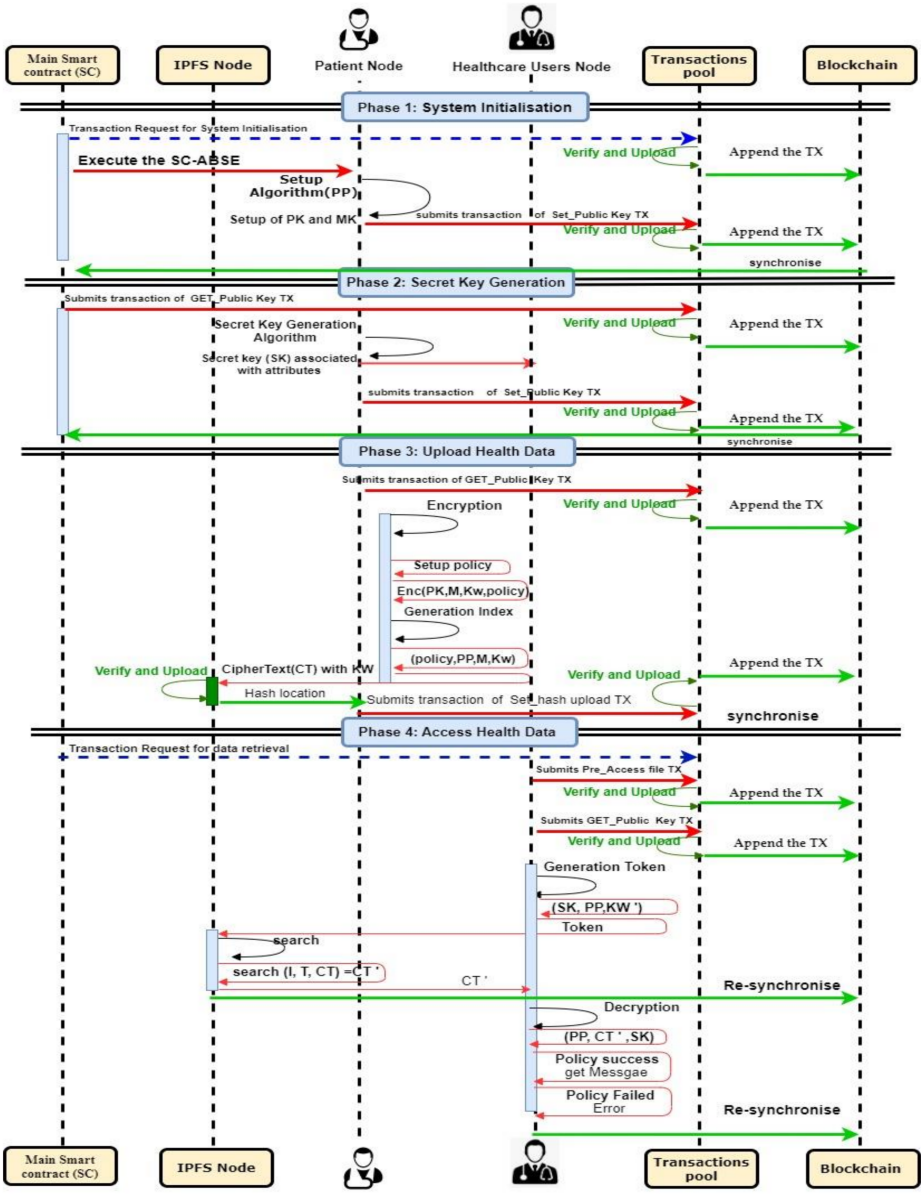


Figure 3. Workflow of generating the smart contract.

4.4. Security Model of SC-ABSE

The proposed scheme needs to satisfy the following security requirements: (1) selective security against CKA and (2) keyword secrecy. The following security game experiment is performed on the interactive play between the adversary (A) and the challenger (C).

4.4.1. Chosen Keyword Attacks

On the basis of the security model in the study [60], the provable security indistinguishability experiment CKA is formalised as $SC - ABSE_{A,\Pi}^{SCKA}(n) = 1$ by the next security game.

1. Setup: The C executes the *Setup* algorithm in the system initialisation phase of the proposed scheme to generate public and master keys. Then, the C sends the public keys to the A and keeps the master keys secret.
2. Query Phase 1: The C establishes the list of the keyword list L_{KW}, whereas all the lists are initially empty. In addition, the A receives a response as a polynomial number of queries from the C as follows:

- SK generation query (A_{sk}): The C executes the *KeyGeneration* algorithm to obtain SK for users in the HUN, where A_{sk} is a set of attributes if $A_{sk} \notin \mathbb{A}$; then, the C outputs $\perp$.
- GenerationToken Query $A_{sk,kw}$: The C executes the *GenerationToken* algorithm to generate T (Token) and send it to the A. Finally, if A_{sk} is satisfied with the access structure policy $\mathbb{A}$, then the C adds the keyword to list L_{KW}.

3. Challenge: The A chooses two equivalent keywords (kw^1, kw^2) in which that $(kw^1, kw^2) \notin$ to the list of the keywords L_{KW}. Afterwards, the C chooses $b \in \{0, 1\}$ and returns over execute the *GenerationToken* algorithm to produce a keyword index and send it to A. Finally, the C sets I_b =*Encryption* and sends it to A.
4. Query phase 2: It is a comparable to query as phase 1. However, the A may continue query towards GenerationToken Query $(A_{d,\,kw})$ keyword except if $A_d \in T$. Otherwise, the A is incapable of querying.
5. Guess: The A guesses $b\ of\ b'$, where $b \in \{0, 1\}$, and it has to be returned by the A. The A wins the experiment game only if $b = b'$, where the output of the game is defined to be 1. Then, we can write $SC - ABSE^{SCKA}_{A,\,\Pi}(n) = 1$. If the output of the game is 0, then the adversary loses the game.

Definition 1. *The proposed scheme can be secure against the indistinguishability of CKA via Equation (3) if all PPT adversaries had a negligible function.*

$$\Pr(SC - ABSE^{CKA}_{A,\,\Pi}(n) = 1 \leq 1/2 + negligible\,(n) \tag{3}$$

4.4.2. Keyword Secrecy

The provable security experiment of keyword secrecy is formalised as $SC - ABSE^{\mathrm{ks}}_{A,\,\Pi}$. The security game of the keyword secrecy (KS) can be proven via the next experiment [60].

1. Setup: The C runs the Setup algorithm in the system initialisation phase of the proposed scheme to generate the public and master keys. Then, the C sends the public keys to the A and keeps the master keys secret.
2. Query Phase 1: The A issues the below algorithms in polynomial times.
   - *SK generation query*(A_{sk}): The C runs the *KeyGeneration* algorithm to obtain SK for users in the HUN, where A_{sk} is a set of attributes if $A_{sk} \notin \mathbb{A}$. Then, the C outputs $\perp$. Otherwise, the C inserts keywords into the list of established medical data queries.
   - *GenerationTokenQuery*$(A_{sk,\,kw})$: The *GenerationToken* algorithm is run by the C to generate T by giving the SK and set of kw. The C sends the token to the A only if the A_{sk}. attribute set satisfies the access structure policy. $\mathbb{A}$.
3. Challenge: The A selects the SK and passes it to the C, whereas the C must choose the keyword set (kw') from the medical data space and run the *Encryption* algorithm to send the index I to the A.
4. Guess: The A guesses the distinguishable keywords by outputting the keyword set (kw'). The A can win a security game by defining the following $SC - ABSE^{\mathrm{ks}}_{A,\,\Pi} =$ 1 experiment if and only if $(kw' = kw)$.

Definition 2. *The proposed scheme can be secure against the indistinguishability of keyword secrecy attack if for all PPT adversary A and the advantage of the adversary to breaking the above keyword secrecy game is at most* $SC - SABE^{ks}_{A,\,\Pi} = (n)$*, and it has negligible probability in security parameter* n.

4.5. Concrete Construction

The proposed SC-ABSE scheme is built on the base of the schemes of [38] and [22]. The notations used in concrete construction are presented in Table 3. The concrete construction of SC-ABSE is described in the following four phases:

Table 3. Notation Definitions.

Notations	Explanation
$\mathbb{G}_1$, $\mathbb{G}_2$	Multiplicative cyclic bilinear groups
$\boldsymbol{h_1}$, $\boldsymbol{h_2}$	Collision-resistant hash functions
$\boldsymbol{g}$	Generator of the group $\mathbb{G}_1$
$\boldsymbol{e}$	Bilinear map
PK	Public key
MK	Master key
SK	Secret key
SKE	Symmetric key algorithm
$\mathbb{A}$	Access structure policy
$\boldsymbol{\mu} = \{\mathbf{1}, \ldots, \mathbf{n}\}$	A number set of the attribute
$\mathbf{MD} = \{\boldsymbol{md_1}, \ldots, \boldsymbol{md_p}\}$	Set of medical files
KW	Set of keywords for medical file
$\mathbf{KW_{file}}\{\boldsymbol{kw_{I_1}}, \ldots, \boldsymbol{kw_{I_m}}\}$	Set of keywords included in medical file
$\mathbf{KW'} = \{\mathbf{kw'_1}; \ldots, \mathbf{kw'_t}\}$	Set of Submitted keyword for medical file
$\boldsymbol{I_j}/\boldsymbol{j} \in \{\mathbf{1}, \ldots, \boldsymbol{m}\}$	$j-th$ keyword in KW_{file}
$\mathbf{I'_i}/\mathbf{i} \in \{\mathbf{1}, \ldots, \mathbf{t}\}$	$i-th$ submitted keyword in KW_{file}
$\lvert\boldsymbol{S}\rvert$	Set of attributes for users in the HUN
T	Token generation query for a conjunctive keyword
I	Index set of conjunctive keywords

4.5.1. System Initialisation Phase

The Setup $\left(1^{\mathbf{k}}, \boldsymbol{\mu}\right)$ algorithm is executed by the PN in the blockchain-based PHR system to generate public security parameters. It defines the attribute space μ, where any $j \in \mu$, and the security parameter 1^{k}. Given two multiplicative cyclic groups $\mathbb{G}^1$ and $\mathbb{G}^2$ of prime order p with generators of g^1 and g^1 for the security parameter of 1^{k}. And then, maps the parameters of bilinear $e : \mathbb{G}^1 \times \mathbb{G}^2 \rightarrow \mathbb{G}^3$ as a tuple of ($\mathbb{G}^1$, $\mathbb{G}^2$,q, g, e). Afterwards, the two random oracle collision-resistant hash functions $h_1 : \{0,1)* \rightarrow \mathbb{G}^1$ and $h_2 : \{0,1)* \rightarrow \mathbb{Z}_q$ are defined. Moreover, it randomly selects the element of $\alpha, \beta, \gamma \in \mathbb{Z}_q$ by computing the following exponents as $X = g^{\alpha}$, $Y = g^{\beta}$, $Z = g^{\gamma}$. The PN in the blockchain conclusively establishes the security parameter of the system by publishing the PK (as in Equation (4)) and keeping the MK as a secret (as in Equation (5)).

$$\mathbf{PK} = \left(\mathbb{G}^1, \mathbb{G}^2, q, g, e, h_1, h_2, X, Y, Z\right) \tag{4}$$

$$\mathbf{MK} = (\alpha, \beta, \gamma) \tag{5}$$

The PN submits a transaction of $(Set_{public\ key}TX)$ into the transaction pool to be validated by the miners. Whereby, $(Set_{Pk}TX = \mathrm{Set}_{\mathrm{Public\ Keys\ (PK)}}, 1^{\mathrm{k}})$ is the format of the above transaction. Once the transaction has been broadcasted on the main blockchain network. The smart contract begins compiling and deploying its particular function, such as generating an SK, encrypted patient medical data, generated tokens, performed searches and decrypted. However, any participating node can submit a $\mathrm{Get}_{(public\ key)}$TX transaction to the main network of the blockchain. Then, the main smart contract invokes the setup algorithm to initialise public security parameters $\left(1^{\mathrm{k}}\right)$.

4.5.2. Secret Key Generation Phase

The *KeyGeneration* $(\boldsymbol{S}, \boldsymbol{MK})$ algorithm starts executing via the PN by selecting a set of attributes S to generate the final SK for authorised users in the HUN. The user sends

the registration request to the PN node to authenticate the user's identity and assigns the appropriate attributes (S) by including the user's account address to the set of permitted users. The PN first chooses a random element of $r \in \mathbb{Z}_q$ and then randomly selects an $r_j \in \mathbb{Z}_q$ for each attribute, whereby $j \in$ to set of the attribute (s). Finally, the PN outputs an SK for different users with Equation (6).

$$\mathbf{SK} = \left(\pi = \mathrm{g}^{(\alpha\gamma - \mathrm{r})/\beta}, \left\{ \lambda_j = h_2 g^r (j)^{r\,j} \mu_j = g^{r\,j} \right\} j \in S \right) \tag{6}$$

The PN submits the following transaction format ($\mathrm{Set}_{\mathrm{sec\,ert\ key}}\mathrm{TX}$ = set of attributes (S), SK) into the transaction pool to issue SK attributes for users in the HUN. Afterwards, the PN invokes a key generation algorithm in the main smart contract to generate an SK and sends the keys to the users in a secure channel. Each user has been recorded securely on the blockchain with a defined unique attribute authorisation.

4.5.3. Upload Health Data Phase

Encryption $(\boldsymbol{PK}, \boldsymbol{MD}, \boldsymbol{kw}, \mathbb{A})$. The users run the algorithm in the PN to establish the keyword index and the encrypt procedure for their medical record set. The algorithm takes PK and the patient health record file for $File = \{f = MD\}$ and the keyword dictionary $KW = \{kw\}$ as input. The procedure for encrypting and uploading patient medical data in main smart contracts is described as follows:

- Step 1: The patient users define a keyword as $KW_{file} = \{kw_{I1}, \ldots, kw_{Ij}, \ldots, kw_{Im}\}$ for each health data file $MD \in f$ and selects random elements $r_1, r_2 \in \mathbb{Z}_q$ to generate index I, where $I_j = 1$ represents the $j - th$ keyword in KW_{file}, where this keyword is embedded in MD.
- Step 2: The patient users randomly select AES symmetric key SKE_{file} from the key space and encrypt each file via Equation (7).

$$\mathrm{FileEnc(f)} = \mathrm{Enc_{AES}} \left\{ file, SKE_{file} \right\} \tag{7}$$

- Step 3: The patient users protect the keyword in the file KW_{file} and the symmetric key encryption of file SKE_{file} under the access policy structure $\mathbb{A}$. Consequently, the users select the polynomial of q_x for every individual node x in the access policy structure $\mathbb{A}$ by starting from the root node r in a top-down manner. For every individual node x, the degree d_x of the polynomial q_x is set as $d_x = k_x - 1$, where k_x is the threshold value of the individual node x. To define the points of the polynomials q_r and q_x fully, the root node r and node x need to start these algorithms as $q_r(0) = r_2$ and $q_x\,(0) = q_{parent\,(x)}(index(x))$, respectively, and then randomly select the elements d_r and d_x. The set of leaf nodes in $\mathbb{A}$ is as l_n, and the index I along with KW_{file} and SKE_{file} are encrypted by giving the tree access structure $\mathbb{A}$ and computing it with $\delta_i = X^{r_1 h_2(kW_i, ske_i)}$ whereby $i \in \left\{1, \ldots, MD_{file}\right\}$. Then, the patient users randomly choose the element of α, β, $\gamma \in \mathbb{Z}_q$ by computing the following exponents as $\mathbb{E}_0 = X^{r_2}$, $\mathbb{E}_1 = Y^{r_2}$, $\mathbb{E}_2 = Z^{r_1}$. Finally, $Protect_{kw_{file}, ske_{file}}$ can be performed by Equation (8).

$$Enc_{\mathbb{A}}\left(KW_{file}, SKE_{file}\right) = \left(\theta_y = h_1(\mathrm{attribute}(l_n))^{q_{\ln(0)}}\right) \tag{8}$$

- Step 4: The patient users upload the file tuple, including encrypted health data $\boldsymbol{MD}^*$, encrypted symmetric key encryption for medical data (SKE_{file}) and the encrypted indexed data I, whereby $I = \{\{\delta_i\}, \{\theta_y\}, \mathbb{E}_0, \mathbb{E}_1, \mathbb{E}_2\}$ to be stored in decentralised storage. IPFS receives the users' tuple file and places the file location $h_{location}$ and then returns the hash value of that file to be stored in the DHT of the PN. Then, the PN submits an unconfirmed transaction $\left(\mathrm{Set}_{(hash_{location})}\mathrm{hash\ upload\ TX}\right)$ to the miner's pool for validation purposes and broadcasts it on the blockchain's main network.

4.5.4. Access Health Data Phase

At this phase, the HUN users need to initialise the following transaction format $\left(\text{Submit}_{\text{transaction}} = \text{Pre}_{(\text{Access file})}\text{TX}\right)$ to the transaction pool to retrieve the patient file. Once the transaction has been approved, users perform token, search and decrypt processing functions in the main smart contract for the medical data stored in IPFS storage.

GenerationToken (SK, S, KW). The users in the HUN creates a token that intends to search for patient data. They first select a random element $s \in \mathbb{Z}_q$ and computes the token $T_1 = \prod_j^t g^{s\alpha h_2(kw_i)}$, $T_2 = g^{s\gamma}$, $T_3 = \pi^s$. Then, they compute $\lambda'_j = \lambda^s{}_j$, $\mu'_j = \mu^s{}_j$ for each attribute $j \in S$ and set the token of the submitted keyword set as $T = \left(s,\ T_1, T_2, T_3, \left\{\lambda'_j, \mu'_j\right\}_{j\in s}\right)$.

Search $(\text{I}, T_{KW},\ S,\ \mathbb{A})$. The IPFS node receives the token and attributes of the users of the HUN. The IPFS searches for keywords if the attributes and tokens are satisfied with the access policy structure embedded into the index keywords and encrypted symmetric key encryption. Then, it returns the relevant tuple file through the following steps:

- Step 1: The leaf node is x, and $j' =$ attribute (x) only if $j' \in S$. Then, it computes MD_x with Equation (9). Otherwise, $MD_x = \perp$.

$$MD_x = \frac{e(\lambda_j',\ \delta_x)}{e(\mu_j',\ \theta_x)} = e(g,\ g)^{rsq_x(0)} \tag{9}$$

- Step 2: The nonleaf node is x, and the arbitrary $k_x - size$ set of children x' is kw_x. If the set of children x' in node x does not exist, then $MD_{x'} \neq \perp$. Otherwise, it calls the recursive algorithm to compute the following: $MD_x = \prod_{x' \in MD_x} MD_{x'}{}^{\Delta_{i,k\omega'_x}{}^{(0)}} = e(g,\ g)^{rsq_x(0)}$, where $i = index(x')$, $k\omega'_x = \{index(x') :\ x' \in kw_x\}$.
- Step 3: IPFS verifies if Equation (10) holds, and then IPFS sends relevant results, including encrypted medical data MD_{kw} containing the keyword kw and corresponding symmetric encryption key SKE_{file}, to users. Otherwise, it returns $\perp$, where $MD_r = e(g,\ g)^{rsq_r(0)} = e(g,\ g)^{rsr_2}$.

$$e\left(\prod_{i=1}^t \delta_i \mathbb{E}_0, T_2\right) = e(\mathbb{E}_2,\ T_1) MD_{r^e}(\mathbb{E}_1, T_3) \tag{10}$$

Decryption $\left(\text{MD}^*, SKE_{file}\right)$: This algorithm is executed by the users of the HUN in accordance with the returned tuple file from the IPFS to obtain the plaintext of the patient file via Equation (11).

$$\text{FileDec(f)} = \text{Dec}_{\text{AES}}\left(\text{MD}^*, SKE_{file}\right) \rightarrow (\text{MD}) \tag{11}$$

4.5.5. Correctness

The demonstration of the proposed scheme SC-ABSE correctness is conducted by proving that the search algorithm procedure is correct if the attributes and tokens are satisfied with the access policy structure embedded in the index keywords and the encrypted symmetric key encryption. The correctness of Equation (10) can be verified as follows:

Let

$$\begin{aligned} e(\mathbb{E}_2,\ T_1) &= e(g,\ g)^{r\alpha\gamma r_1 \sum_{i=1}^t h_2{}^{(kw'_i)}} \\ e(\mathbb{E}_1,\ T_3) &= e(g,\ g)^{r\alpha\gamma r_2 -\ rsr_2} \\ M_r &= e(g,\ g)^{rsr_2} \end{aligned} \tag{12}$$

and

$$e(\mathbb{E}_2,\ T_1) = M_r e(\mathbb{E}_1,\ T_3) = e(g,\ g)^{a\alpha\gamma(r_2 +\ r_1\ \sum_{i=1}^t h_2{}^{(kw'_i)})} \tag{13}$$

Then,

$$e\left(\prod_{i=1}^{t}\delta_i\mathbb{E}_0, T_2\right) = e\left(g^{a\alpha\gamma(r_2+\ r_1\ \sum_{i=1}^{t} h_2{}^{(kw'_i)})}\right)\ g^{s\gamma},$$
$$= \ a\alpha\gamma\left(r_2 + \ r_1\ \sum_{i=1}^{t} h_2{}^{(kw'_i)}\right) \tag{14}$$

Therefore, Equation (10) holds that

$$e\left(\prod_{i=1}^{t}\delta_i\mathbb{E}_0, T_2\right) = e(\mathbb{E}_2,\ T_1)M_{r^e}(\mathbb{E}_1, T_3). \tag{15}$$

5. Security Analysis

This section discusses the security analysis of the proposed SC-ABSE scheme to validate its robustness in meeting various security requirements. The analysis of security validation is divided into two parts. The first part conducts a formal security analysis on the basis of the DBDH hardness assumptions and DL problems. The second part utilises AVISPA to verify the security correctness of the proposed scheme.

5.1. Semantic Security Proof

The security aspects of the proposed SC-ABSE scheme are established on the basis of the following theorems. The security proofs of theorems 1 and 2 are comparable to the scheme in [61].

Theorem 1. *Suppose the DBDH problem is hard relative to $\mathcal{B}$. In that case, the proposed SC-ABSE scheme, under which access control has been built, is secure against the standard model's CKA.*

Proof. Let Π be donated to the proposed SC-ABSE scheme and A be donated to the PPT in the $SC - ABSE^{\mathrm{CKA}}_{A,\ \Pi}(n)$ security game experiment referred to in Section 4.4. Construct a simulator A^* an adversary that acts as an adversary trying to solve the DBDH problems. The A^* chooses four random elements $\left(n, p, \mathbb{G}_1, \mathbb{G}_2, e, g, g^{\alpha}, g^{\beta}, g^{\gamma}, e(g,g)^{z}\right)$, where $\mathcal{B}(1^n) \rightarrow ((n, p,\ \mathbb{G}_1\ \mathbb{G}_2\ , e)$ and $\mathrm{g} \in \mathbb{G}_1$ and $\alpha,\ \beta,\ \gamma \in \mathbb{Z}_p$, whereas $z = \alpha,\ \beta,\ \gamma$ is the random element of $\mathbb{Z}_p$. The capability of A^* is to determine the unknown value of z. In the construction of the security game of the SCKA, the A^* executes A as follows:

1. Setup: The A selects an access structure policy $\mathbb{A}$ and submits it to A^*. Then, the A^* selects a consistent element $h_2 \in \mathbb{G}_1$ and presumes $t, s\ \in\ \mathbb{Z}_p$ by setting the parameter as follows:

$$(g_0 = g), \tag{16}$$
$$(g_1 = g^{\alpha}), \tag{17}$$
$$(g_2 = g^{t}), \tag{18}$$
$$\left(h = g^{\beta}\right), \tag{19}$$
$$(h_1 = g^{\alpha}g^{-s} = g^{\alpha-s}), \tag{20}$$
$$h_2 = e\left(h^{-1}, h_1\right). \tag{21}$$

The A^* sends the $(n, \mathbb{G}_1, \mathbb{G}_2, e, g_0, g_1, g_2, h, h_1, h_2)$, $\left\{pk_j = g^{sk_j}\right\}_{j\in\mathbb{A}}$ to the A, where the SK $sk_j \in\ \mathbb{Z}_p$ has been selected randomly for each $j\ \in\ \mathbb{A}$. Then, by comparing Equations (4) and (18), the trivial value of g^t is found to be equal to β. In addition, presume that for unknown $g_3 \in\ \mathbb{G}_1$, $h_2 = g_3{h_1}^{\beta=t} = g_3{h_1}^{t}$ in Equation (5) of MK generation, which concludes that the PK has been correctly chosen.

2. Query Phase 1: The A establishes the list of L_{KW}, and the A^* keeps a list of the L_{KW} for each data user, whereas all the lists are initially empty. In addition, the A^* receives a response as a polynomial number of queries from A as follows:
   - *SK generation query*($\boldsymbol{A_{sk}}$): The A runs the *KeyGeneration* algorithm to generate SK for the users in the HUN. Then, the A^* checks whether the set of an attribute in A_{sk} satisfies policy $\mathbb{A}$ or not; if not, then for any attribute $j \in$ s, it is set as in Equation (22), and then the A returns $\boldsymbol{A_{sk}} = \{Sk_j\}$ **,** where $j \in$ to set the attribute (s) towards the A^*.
   $$sk_j = h_2 g^{s\beta=sk_j} \tag{22}$$
   - GenerationToken Query ($\boldsymbol{A_{sk,\, kw}}$): It is comparable to a query of *KeyGeneration*. The A^* has to generate $\boldsymbol{A_{sk}} = \{\boldsymbol{Sk_v}\}$ **,** where $j \in$ attributes. The A^* needs to select a random element of $s \in \mathbb{Z}_p$ by creating the search token via Equation (23) and checks whether the attribute satisfies $\mathbb{A}$ or not; if yes, then A^* adds the keywords to list L_{KW} by sending to the A.
   $$T = \left(T_1 = \prod_j^t g^{s\alpha \mathrm{h}_2(kw'_i)}, T_2 = g^{s\gamma}, T_3 = \pi^s, \left\{\lambda'_j, \mu'_j\right\}_{j\in s}\right) \tag{23}$$

By recalling Equations (6), (17) and (20), the SK generated in Equation (22), and thus the search token in (23) is valid as follows:

$$h_2 g^{s\beta=sk_j} = g_3 h_1{}^{\beta=t} \overset{(20)}{=} g_3 \left(g^{\alpha-s}\right)^{\beta} g^{s\beta} sk_j = g_3\, g^{\alpha\beta} sk_j \overset{(17)}{=} g_3\, g^{\alpha\beta} sk_j{}^{g^t=\beta}_{=}\, g_3\, g_1^t sk_j \overset{(6)}{=} sk_j \tag{24}$$

3. Challenge: The A returns two equivalent keywords $\left(kw^1, kw^2\right)$ to the A^* in which $\left(kw^1, kw^2\right) \notin$ to the list of the keywords L_{KW}. In addition, the A^* selects a random element of $r_1, r_2 \in \mathbb{Z}_q$ to generate encrypted index I. Then, the A^* gives the encrypted index $I(r_1, r_2)$, keyword health data file $KW_{file} = \left\{kw_{I1}, \ldots, kw_{Ij}, \ldots, kw_{Im}\right\}$, and public key PK to the A. Then, the A^* must select the random value of $b \in \{0, 1\}$ by assuming that if and only if $\mathbb{Z}_q = \alpha,\ \beta,\ \gamma$ and the element of encrypted index $I(r_1, r_2)$ in the encryption algorithm presented in Section 4.5.3 are selected randomly and correctly.
4. Query phase 2. The A continues his query to the oracles, and the A^* responds to a query that is identical to the procedure in phase 1.
5. Guess: The A guesses $b\ of\ b'$, where $b \in \{0, 1\}$. The simulator A^* checks whether $b = b'$ or not. If yes, then the output is equal to 1; otherwise, the output is 0.

As shown in the challenge step, if $\mathbb{Z}_q = \alpha,\ \beta,\ \gamma$ is selected correctly, then the responses returned to A are valid, and the output is as follows:

$$\Pr\left(A^*\left(n, p, \mathbb{G}_1, \mathbb{G}_2, e, g, g^{\alpha}, g^{\beta}, g^{\gamma}, e(g,g)^{\alpha,\,\beta,\,\gamma}\right) = 1\right) = \Pr\left(SC - ABSE^{\mathrm{CKA}}_{A,\,\Pi}(n) = 1\right) \tag{25}$$

On the contrary, if (r_1, r_2) is a random element of $\mathbb{Z}_q$, then the keyword is also a random element of $\mathbb{G}_2$. No leakage of any information occurs on the kw_i or the analogue. In addition, the A cannot learn any partial information about i, and the output is as follows:

$$\Pr\left(A^*\left(n, p, \mathbb{G}_1, \mathbb{G}_2, e, g, g^{\alpha}, g^{\beta}, g^{\gamma}, e(g,g)^{z}\right) = 1) = 1/2\right) \tag{26}$$

By combining Equations (25) and (26), however, the assumption states that the DBDH is hard relative to $\mathcal{B}$. Then, a negligible function exists such that

$$\begin{gathered}\left|\Pr\left(SC - SABE^{\mathrm{CKA}}_{A,\,\Pi}(n) = 1\right) - 1/2\right|, = \\ \left|\Pr\left(A^*\left(n, p, \mathbb{G}_1, \mathbb{G}_2, e, g, g^{\alpha}, g^{\beta}, g^{\gamma}, e(g,g)^{\alpha,\,\beta,\,\gamma}\right)\right. = 1 - \right. \\ \left.\left.\Pr\left(A^*\left(n, p, \mathbb{G}_1, \mathbb{G}_2, e, g, g^{\alpha}, g^{\beta}, g^{\gamma}, e(g,g)^{z}\right) = 1\right)\right)\right| \leq \text{ negligible (n)},\end{gathered} \tag{27}$$

thus proving the theorem. □

Theorem 2. *Suppose the DBDH problem is hard relative to $\mathcal{B}$. In that case, the proposed SC-ABSE scheme, under which access control was built, gains the keyword secrecy (KS) in the standard model.*

Proof. In the $SC-ABSE_{A,\,\Pi}^{\mathrm{CKA}}(n)$ security game experiment in Section 4.4, let Π be donated to the proposed SC-ABSE and the A and the A^* donated to the PPT, whereas the assumption of the problem is DL. The authors presume that the elements of $\left(n, p, \mathbb{G}_1\, \mathbb{G}_2\, , e, g, g^{kw}\right)$ have been given to the A^*, where n is the system's security parameter and $\mathcal{B}(1^n) \rightarrow (n, p, \mathbb{G}_1\, \mathbb{G}_2\, , e)$ is the relative function, where $g \in \mathbb{G}_1$, and $kw \in \mathbb{Z}_p$ is used. The mission of A^* is to calculate the trivial value of kw. The A^* executes A in the security adversary model of the keyword secrecy as follows:

1. Setup: The A^* chooses universal attribute set and random elements $\alpha,\ \beta,\ \gamma\ \in \mathbb{Z}_p$. Then, the A^* generates the public parameters of the system $= (n, \mathbb{G}_1, \mathbb{G}_2, e, g_0, g_1, g_2, h, h_1, h_2)$, $\{pk_j\}_{j\in\mathbb{A}}$, and $mk = \alpha,\ \beta,\ \gamma$ by executing the Setup $\left(\mathbf{1^k}\right) \rightarrow \boldsymbol{Pk}, \boldsymbol{MK}$ algorithm. Then, the A^* gives the pk to A and keeps the MKs secret.
2. Query phase 1: The A makes a query to the oracle *SK generation Query* (A_{sk}). Then, the A^* executes the *KeyGeneration* algorithm to generate $\boldsymbol{SK}$ for the users in the HUN. Finally, the A^* gives $\boldsymbol{SK}$ to the A.
3. Challenge: The A declares that Phase 1 is over. Then, the A^* runs the *KeyGeneration* algorithm to generate $\boldsymbol{SK}$ and selects a random element of $r_1, r_2\ \in\ \mathbb{Z}_q$ to generate index I. Then, A^* sets the keyword by $kw = \delta_i =\ X^{r_1 h_2(kW_i, sk_i)}$ whereby $i \in \left\{1, \ldots,\ MD_{file}\right\}$. Then, A^* returns the kw, I and the PK to the A.
4. Guess: The A guesses the distinguishable keywords kw'.

If A wins the game experiment of $SC-ABSE_{A,\,\Pi}^{\mathrm{ks}}$, then the A^* can solve the DL problem. In addition, the output of the experiment is as follows:

$$\Pr\left(A^*\left(n, p, \mathbb{G}_1\, \mathbb{G}_2\, , e, g, g^{kw'}\right) = kw\right) \geq \Pr\left(SC-ABSE_{A,\,\Pi}^{\mathrm{ks}} = 1\right) \tag{28}$$

On the contrary, if the game experiment is under the hardness assumption of the DL problem is believing to says the security game of $SC-ABSE_{A,\,\Pi}^{\mathrm{ks}}$ is

$$\Pr\left(A^*\left(n, p, \mathbb{G}_1\, \mathbb{G}_2\, , e, g, g^{kw'}\right) = kw\right) \leq \text{negligible (n)} \tag{29}$$

By combining Equations (28) and (29), however, the probabilistic of $SC-ABSE_{A,\,\Pi}^{\mathrm{ks}} = 1$ is a negligible function in n of the parameter security, thus proving the theorem. □

Theorem 3. *Suppose SC-ABSE = $\prod(X)$commits to enforce that the patient's medical data stored in the IPFS cannot be interfered with or changed. In that case, the proposed SC-ABSE scheme, under which access control was built, manages to gain tamper-proof medical data stored in IPFS.*

Proof. The proposed SC-ABSE = $\prod(X)$ scheme gains tamper-proof medical data stored in IPFS from the features of the blockchain. It starts by encrypting personal health data with a traditional symmetric encryption algorithm, such as AES. It then uploads the encrypted medical data to the IPFS storage that cannot be modified because upon receiving the patient's tuple file, the IPFS has to locate an appropriate location for that file, return the hash value $h_{location}$ of that file to the PN to store it securely, submit the transaction $\left(\mathrm{Set}_{(hash_{location})}\text{hash upload TX}\right)$ to the miner's pool for validation purposes and upload it to the main blockchain network. The moderate characteristics of the tamper-proof blockchain ensure the integrity of the data in the transaction and prohibit tampering and failure at any point unless one has upwards of 51% of the computational power of the entire blockchain network, thus proving the theorem. □

Theorem 4. *Suppose SC-ABSE = $\prod(X)$ can ensure that users in the HUN comply with the access structure to retrieve patient data stored in the IPFS node even if they combine token generation. In that case, the proposed SC-ABSE scheme, under which access control was built, manages to achieve resistance to the collusion attack.*

Proof. The proposed SC-ABSE = $\prod(X)$ is collusion resistant. In the key generation phase of the proposed scheme, the PN requests that users in the HUN delegate their own private key account to be authenticated to assign appropriate attributes and then add the valid user account address of the medical practitioner to the smart contract collection of authorised users. Moreover, the PN has to select a random element $r \in \mathbb{Z}_q$ and then randomly select a $r_j \in \mathbb{Z}_q$ for each attribute by which $j \in$ to set of attributes and insert it into the SK attribute for users that were performed through Equation (6). To validate the users, an appropriate token generation of T must be generated using its own sk so that the IPFS storage node can obtain $T_1 = \prod_j^t g^{s\alpha h_2(kw'_i)}$, $T_2 = g^{s\gamma}$, $T_3 = \pi^s$, where $e(g,g)^{s\alpha h_2}$. Then, IPFS must perform the search algorithm correctly. Otherwise, if invalid users tend to combine their own private keys to form an SK. Therefore, the IPFS node cannot obtain $e(g,g)^{s\alpha h_2}$ in which the random elements are inserted in the private keys of users in the HUN, thus proving the theorem. □

5.2. Security Validation of the Proposed SC-ABSE Scheme in AVISPA

This section presents the security properties of the proposed SC-ABSE scheme by using the AVISPA tool [62]. AVISPA is an automatic tool for evaluating the feasibility of secure internet protocols and applications. This tool offers an expressive and modular formal language for defining protocols and their security properties by incorporating different back-end automated analysis techniques [63]. Many researchers have widely adopted this automatic security verification to validate the security of their proposed schemes [64–67]. SC-ABSE was designed and coded by a specific programming language called the high-level protocol specification language (HLPSL) via the animator's security protocol (SPAN) [68]. AVISPA starts translating the high-level language into the intermediate format's lower-level language (IF) with the built-in translator features named hlpsl2if. Subsequently, the IF specification executes the four backends of the AVISPA tool, namely, the on-the-fly model checker (OFMC), the constraint logic-based attack searcher (CL-AtSe), the SAT-based model checker (SATMC) and the tree automated protocol analyser (TA4SP). AVISPA can then automatically output the scheme's analysis result on the basis of whether the security requirements are satisfied or violated. Nevertheless, the SATMC and TA4SP backends are less utilised for security protocol validation due to its built-in features that are incapable of verifying algebraic properties, such as modular exponents and XOR operator. This tool mainly strives to validate the protocol security goals against various active and passive attacks, such as MIM and replay attacks.

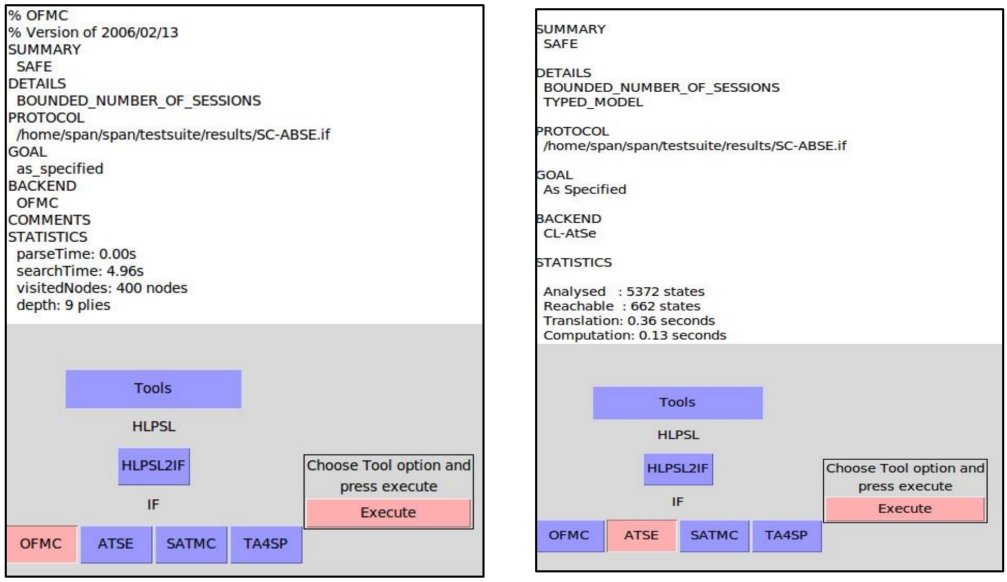
In the security validation of the proposed scheme, the blockchain entity merely establishes and registers each node entity, deploying the smart contract and auditing purposes of all records requests and access activities. The transaction pool contains the unconfirmed transactions of upload and access medical data stored in the IPFS. These two entities considered a constant environment network to deploy such any decentralised applications. Therefore, the security simulation considered three main entities, such as PN, IPFS node and HUN by defining their specifications in the system via HLPLS codes. Moreover, the particular HLPSL code specification, session and environment roles are attached in Appendix A. In Figure A1, the PN enrollment in the blockchain-based PHR is responsible for initialising the system and authenticating each user in the HUN. It also encrypts the medical records and uploads it to the IPFS node. However, the PN needs to communicate with the IPFS node to upload the tuple file ($MD^*SKE_{file}I$). In Figure A2, the users in the HUN need to authenticate the PN to obtain their SK associated with appropriate attributes defined by the users' enrollment. It also communicates with the IPFS by sending a token associated with an interest keyword to retrieve the tuple file ($MD^*SKE_{file}I$). Figure A3,

the IPFS received the encrypted medical data records as a tuple of the file $(MD^*SKE_{file}I)$ need to securely stored file therein. The IPFS also requires to search for the medical records after successfully receiving the token from the HUN. These entities communicate with each other through a secure channel. Substantial consideration must be taken to ensure no secrecy is revealed or passive and active attacks occur.

In addition, Figure A4 presents the specification of the session's role for all main entities by defining its composition. The environment's role is shown in Figure A5 by specifying all essential components within the communication environment, such as symmetric keys, hash function, public key and intruder knowledge. In a simulated environment, the intruder plays a crucial role by replacing the PN, the HUN and the IPFS node roles in the respective sessions in which it intends to compromise the simulated system. The considerations of the secrecy and authentication goals are described in the following aspects:

- *"secrecy_of sec_1"*: The user in the HUN communicates with the PN to obtain the SK and sends it via a secure channel.
- *"secrecy_of sec_2"*: The PN node communicates with the IPFS node and sends a tuple of encrypted medical data in a secure channel.
- *"secrecy_of sec_3"*: The users in the HUN communicates with IPFS and sends a token to IPFS via a secure channel to retrieve the medical data.
- *"authentication_on patient_ipfsnode_tta,tsv,tidi"*: The users in the PN authenticate with IPFS node to upload the file.
- *"authentication_on healthcareusersnode_ipfsnode_tta,tsv,tidi"*: The users in the HUN authenticate with the IPFS node to retrieve the medical data.
- *"authentication_on patient_healthcareusersnode_tsn"*: The PN authenticates the users in the HUN with an appropriate attribute to generate their SK.
- *"authentication_on patient_ipfsnode_ti, r1"*: A request to upload the tuple file is made.

The proposed scheme is simulated with OFMC, and CL-AtSe backends with a limited range of session model checks after the communication sessions have been established in the environmental role. In the case of a replay attack, the CL-AtSe backend checks whether a legitimate agent can execute the specified protocol by searching for a passive intruder. Afterwards, the intruder shares the knowledge of some normal sessions between the legitimate agents. By contrast, the Dolev–Yao attack model is executed by the OFMC backend to check whether an MIM attack is possible. The intruder may have the possibility of intercepting, analysing and altering messages as long as he knows the required keys and send them to anyone else in the name of any other agent. The present analysis outcome reveals that our scheme could hold out against various attacks, such as replay and MIM attacks, and the intended security objective are all satisfied, as shown in Figure 4. Therefore, the proposed SC-ABSE scheme is secure under AVISPA simulation or its equivalent.



(**a**) (**b**)

Figure 4. Verification results obtained from AVISPA: (**a**) on-the-fly model checker (OFMC) backend, (**b**) constraint logic-based attack searcher (CL-AtSe) backend.

6. Performance Analysis

A series of simulations are conducted to evaluate the performance of the proposed scheme. The analysis of performance is divided into two parts. The first part investigates the performance evaluation between the relevant schemes in terms of computational complexity in overhead computation, storage costs and overhead communication. The second part demonstrates the processing time of the smart contracts to deploy the proposed scheme in terms of throughput and latency transaction network.

6.1. Computational Complexity Analysis

This section presents a performance analysis of the proposed SC-ABSE scheme in comparison with the existing relevant CP-ABSE schemes. Table 4 lists the notations used in the theoretical asymptotic analysis. The theoretical analysis of computational costs was performed in the context of asymptotic execution complexity. Four types of computational operations, such as paring operation p^T, exponential operation of e_1^T, e_2^T in $|\mathbb{G}^1|$, $|\mathbb{G}^2|$, respectively, and the hash operation h^T mapping of the element in $\{0,1\}$ to $|\mathbb{G}^1|$, were considered for measuring asymptotic execution complexity. In addition, the theoretical asymptotic storage cost was undertaken in three groups $|\mathbb{G}^1|$, $|\mathbb{G}^2|$ and $|\mathbb{Z}_q|$ based on element size. Experimental analysis was conducted on several tests to evaluate the overall performance. Given that electronic medical datasets are not available to the public, we used real-world Enron datasets, which are widely used in many searchable public-key encryption schemes [69]. The public Enron datasets contain approximately 500,000 e-mails from 150 users distributed in 3500 folders, and its size is approximately 0.5 MB of the message. The experimental simulations using Type-A pairings built over the curve $e(f_q) : y^2 + x^3 + x$ [70]. The group $|\mathbb{G}^1|$, $|\mathbb{G}^2|$ of order p as a subgroup of $e(f_q)$ is a large prime number in the Python pairing-based cryptography (pyPBC) library, where the parameters $p = 160$ bits and $q = 512$ bits [71]. Whereby, the value of the $|\mathbb{Z}_q| = 160$ bits, and $|\mathbb{G}^1| = |\mathbb{G}^2| = 1024$ bits. The access control policies are being assumed in the AND gate configuration. The experimental workstation is implemented on an Ubuntu 18.04.4 LTS with an Intel Core i5 Processor 2.3 GHz and 8.00 GB. We choose 10,000 files from the public Enron datasets and set the number of attributes [1,50] and perform experimental tests over 100 times in accordance with previous schemes.

Table 4. Notations used in asymptotic analysis.

Number	Notations	Explanation
1	$\|N_{att}\|$	Number of medical attributes or data users
2	$\|N_{\mathbb{A}}\|$	Number of leaf nodes in an access tree $\mathbb{A}$
3	e_1^T, e_2^T	Exponential operation time in $\mathbb{G}^1$ and $\mathbb{G}^2$
4	p^T	Pairing operation time
5	h^T	Hash operation time
6	$\|\mathbb{G}^1\|$, $\|\mathbb{G}^2\|$	Length of an element in $\mathbb{G}^1$ and $\mathbb{G}^2$
7	$\|\mathbb{Z}_q\|$	Length of an element in $\mathbb{Z}_q$
9	SKE	Symmetric key algorithm

6.1.1. Computation Overhead

The theoretical asymptotic analyses of computation overhead for schemes are compared in Table 5. The proposed scheme outperforms the other schemes in terms of execution time overhead to generate a secret key for HUN users in the key generation phase performed by the PN. In addition, the overhead execution time for users in the HUN to generate search tokens for accessing health data phases is remarkably reduced in comparison with [55–57]. Moreover, the PN users who generate searchable ciphertext during the upload phase of health data have substantially lower encryption times than other schemes. By contrast, the execution time of the proposed SC-ABSE is higher than that of the scheme in [55] and slightly lower than that of the scheme in [56,57] in the access to health data phase when executing the search algorithm in the IPFS node.

Table 5. Comparison of computation overhead.

Scheme	Key Generation	Searchable Ciphertext Generation	Search Token Generation	Data Retrieving	Decryption
[55]	$e_1^T(2\|N_{att}\|+4)+p+e_2^T$	SKE + $(5\|N_{\mathbb{A}}\|+2)h^T\|N_{\mathbb{A}}\|e_1^T+e_2^T$	$e_1^T(2\|N_{att}\|+3)h^T\|N_{\mathbb{A}}\|$	$(5p+e_2^T)(N_{\mathbb{A}}+3p)\|N_{att}\|e_1^T$	$(e_2^T+SKE)(\|N_{att}\|e_1^T)+2\|N_{att}\|p$
[58]	$e_1^T(2\|N_{att}\|+1)+p+2e_2^T$	$e_1^T\|N_{\mathbb{A}}\|+(2\|N_{\mathbb{A}}\|+1)e_2^T$			$\|N_{att}\|e_2^T+3\|N_{att}\|p$
[56]	$e_1^T(3\|N_{att}\|+2)+p+2e_2^T$	$e_1^T\|N_{\mathbb{A}}\|+e_2^T(3\|N_{\mathbb{A}}\|+1)$	$e_1^T(2\|N_{att}\|+3)\,2e_2^T$	$e_2^T+e_1^T(2\|N_{att}\|+4)p$	$\|N_{att}\|e_2^T+3\|N_{att}\|p$
[57]	$e_1^T(3\|N_{att}\|+3)+p+e_2^T(2+\|N_{att}\|)$	$e_1^T\|N_{\mathbb{A}}\|+e_2^T(2\|N_{\mathbb{A}}\|+4)$	$e_1^T(2\|N_{att}\|+4)\,2e_2^T$	$e_2^T+(2\|N_{att}\|+4)p$	$e_2^T(2+2\|N_{att}\|)p\,e_1^T$
Ours	$e_1^T(\|N_{att}\|+1)$	SKE + $e_1^T(2\|N_{\mathbb{A}}\|+1)h^T\|N_{\mathbb{A}}\|$	$3e_1^T+1$	$(5p+e_2^T)(\|N_{\mathbb{A}}\|+p)$	SKE

The comparison of the actual execution time for the schemes is demonstrated in Figure 5 by setting the number of attributes to 50. In Figure 5a, the key generation algorithm of the proposed scheme takes approximately 13.26 ms in $|\mathbb{G}1|$, whereas other schemes require 301.91 [55], 541.32 [58], 615.46 [56] and 838.18 ms [57]. Given the use of a few pairing operations and requires only one exponential operation of e_1^T in $|\mathbb{G}1|$. By contrast, other schemes need to map two pairing operations to generate an SK. The encryption time versus access policy for the number of leaf nodes used to generate encrypted medical data for 10,000 files needs approximately 437.84s, whereas the other schemes [55,56,58], and [57] take approximately 1010.95, 1117.80, 1248.17, and 1197.76 s, respectively, as shown in Figure 5b, because the users in the PN encrypt the medical data via the one-time operation encryption algorithm AES and then only encrypt the symmetric key file and the keyword via ABE. In Figure 5c, the computational cost of the token generation algorithm in the proposed scheme needs approximately 136.17 ms, whereas other schemes in [55–57] cost approximately 804.96, 448.97 and 843.25 ms, respectively, because the token generation algorithm of the proposed SC-ABSE scheme is not dependent upon the access control policy's number of attributes. In addition, most of the computational complexity tasks of pairing operations have been transferred to the search algorithm. However, in Figure 5d, the search algorithm's performance executed in the IPFS node of the proposed scheme costs approximately 881.56 ms. By contrast, the search time for the schemes in [55–57], is 239.32, 1060.95 and 1099.71 ms, respectively. Figure 5d plots the execution time of the decryption algorithm in the proposed SC-ABSE scheme, which needs approximately

113.03 ms, whereas the schemes in [55,56,58], and [57] cost approximately 319.03, 1044.46, 1055.22, and 1132.92 ms, respectively. Given that medical data decryption does not depend on the number of attributes in the access control policies. By contrast, other schemes need to map two pairing operations based on the number of attributes embedded into encrypted medial data.

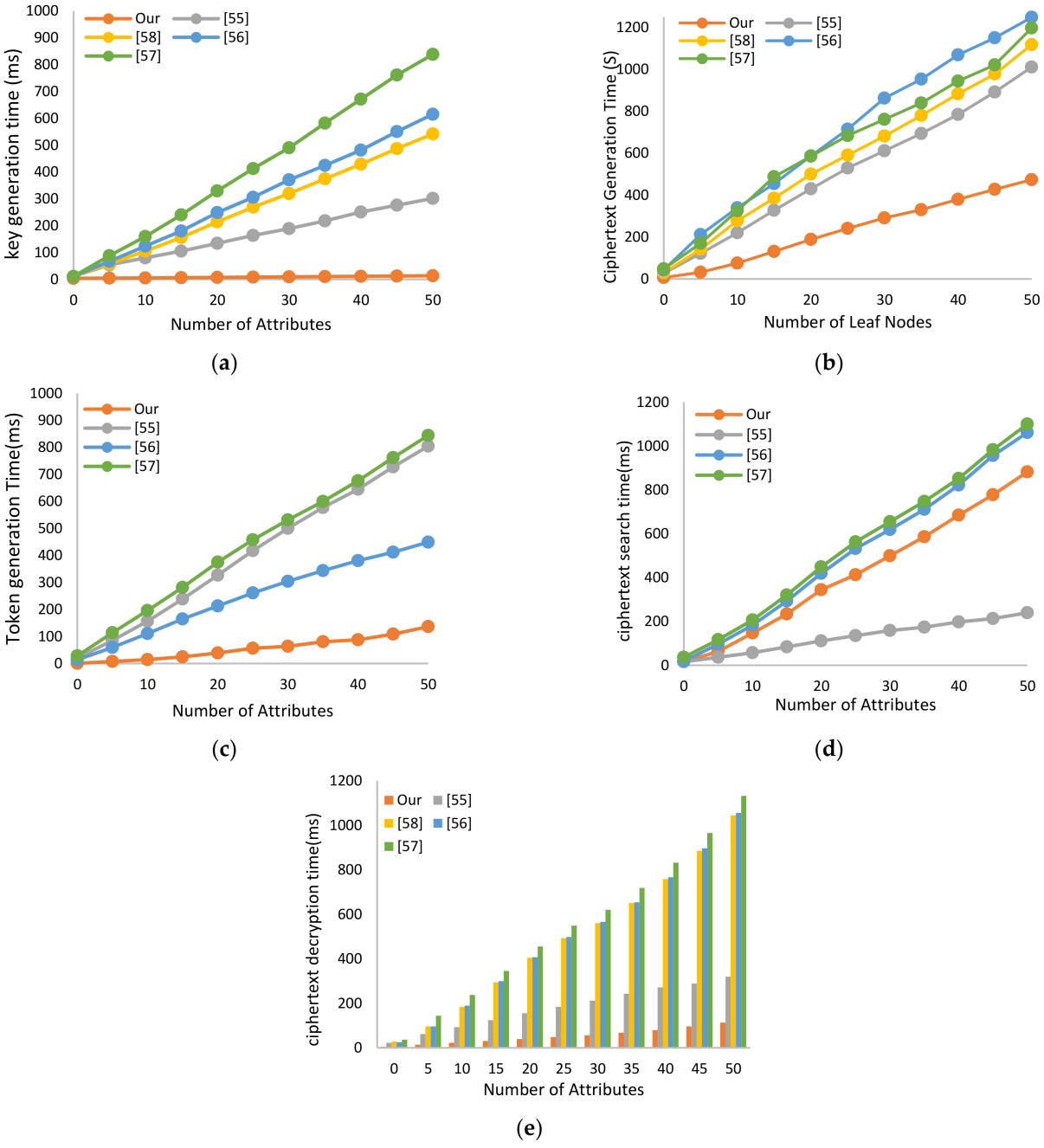


Figure 5. Computation costs: (**a**) execution time for generating users' SK in the patient node (PN) node; (**b**) execution time for users to generate searchable ciphertext in the PN; (**c**) execution time for users in the HUN to create token search; (**d**) IPFS node to retrieve a requested ciphertext; (**e**) execution time in the decryption algorithm.

6.1.2. Storage Cost and Communication Overhead

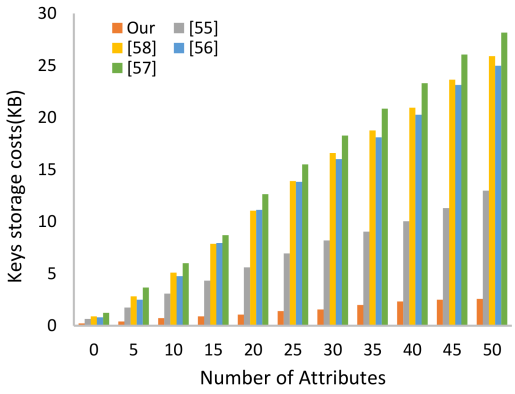
The proposed SC-ABSE scheme is superior to other schemes in terms of storing the SKs of users in the HUN. In addition, the cost of storing one searchable ciphertext in the IPFS node is remarkably decreased by more than half compared with the other schemes.

The communication overhead analysis for transferring the search token from users in the HUN to the IPFS node is dramatically reduced in the proposed SC-ABSE scheme in comparison with others. Therefore, these facts can also be obtained from the data presented in Table 6.

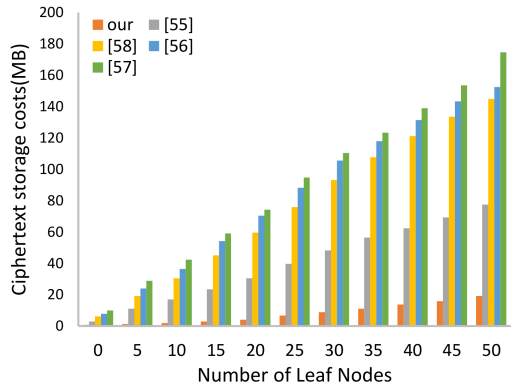
Table 6. Storage costs and communication overhead comparison.

Scheme	Key Length	Searchable Ciphertext Length	Search Token Length
[55]	$(\lvert\mathbb{G}^1\rvert + \lvert\mathbb{G}^2\rvert)(2\lvert N_{att}\rvert + 3)$	$(\lvert\mathbb{G}^1\rvert + \lvert\mathbb{G}^2\rvert)(5\lvert N_{\mathbb{A}}\rvert + 2)\lvert\mathbb{Z}_q\rvert$	$\lvert\mathbb{G}^1\rvert(2\lvert N_{att}\rvert + 3)\lvert\mathbb{Z}_q\rvert\lvert N_{\mathbb{A}}\rvert +$
[58]	$\lvert\mathbb{G}^1\rvert(2\lvert N_{att}\rvert + 1) + \mathbb{Z}_q + 2\lvert\mathbb{G}^2\rvert$	$\lvert\mathbb{G}^1\rvert N_{\mathbb{A}} + \lvert\mathbb{G}^2\rvert(2\lvert N_{\mathbb{A}}\rvert + 1)$	———
[56]	$\lvert\mathbb{G}^1\rvert(4\lvert N_{att}\rvert + 2) + \mathbb{Z}_q + 2\lvert\mathbb{G}^2\rvert$	$\lvert\mathbb{G}^1\rvert\lvert N_{\mathbb{A}}\rvert + \lvert\mathbb{G}^2\rvert(3\lvert N_{\mathbb{A}}\rvert + 1)$	$\lvert\mathbb{G}^1\rvert(2\lvert N_{att}\rvert + 3)2\lvert\mathbb{G}^2\rvert$
[57]	$\lvert\mathbb{G}^1\rvert(4\lvert N_{att}\rvert + 2) + \mathbb{Z}_q + 2\lvert\mathbb{G}^2\rvert(2 + \lvert N_{att}\rvert)$	$\lvert\mathbb{G}^1\rvert\lvert(N_{\mathbb{A}}\rvert + \lvert\mathbb{G}^2\rvert(2\lvert N_{\mathbb{A}}\rvert + 4)$	$\lvert\mathbb{G}^1\rvert(2\lvert N_{att}\rvert + 4)2\lvert\mathbb{G}^2\rvert$
Our	$(\lvert N_{att}\rvert + \mathbb{G}^1)$	$\mathbb{G}^1(2(\lvert N_{\mathbb{A}}\rvert + 1) + 2\mathbb{G}^2$	$3(\mathbb{G}^1 + 1)$

Figure 6 presents the actual performance by setting 50 attributes. In Figure 6a, the storage cost of the SKs of users in the HUN is approximately 2.57 KB, whereas those of the schemes in [55,56,58], and [57] are 12.97, 25.89, 24.97 and 28.15 KB, respectively. In the IPFS node, storage overhead for storing one searchable ciphertext requires approximately 19.08 MB. By contrast, the other schemes [55,56,58], and [57] require approximately 77.43, 144.91, 152.47, and 174.59 MB, respectively, as shown in Figure 6b. In addition, the communication overhead for transferring a search token from the users of the HUN to the IPFS node is approximately 3.71 KB. Conversely, the other schemes in [55–57], cost approximately 27.14, 13.49, and 28.07 KB, respectively, as presented in Figure 6c.

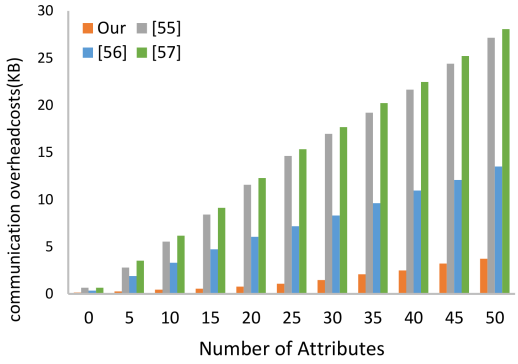


(**a**)



(**b**)

Figure 6. *Cont*.



(**c**)

Figure 6. (**a**) Storage overhead for storing the SKs of users in the HUN; (**b**) storage overhead for storing one searchable ciphertext in the IPFS node; (**c**) communication overhead for transferring a search token from the users in the HUN to the IPFS node.

6.2. Blockchain Network Simulation

This section summarises the proposed scheme's findings deployed on the blockchain network simulation in terms of transaction throughput and transaction latency metrics.

6.2.1. Simulation Setup

The structure of the blockchain network simulation is shown in Figure 7. The network simulation was set up with four nodes on the basis of the Ethereum blockchain docker [72]. All these nodes have been connected to each other on a different virtual machine. Each machine has an Ubuntu 14.04 LTS operating system with 1.6 GHz vCPUs and 2 GB of RAM. The Clique proof-of-authority consensus protocol[7] was used to validate and sign the transaction between the node [73]. However, the consensus mechanism is not in the scope of this discussion. The proposed SC-ABSE scheme alongside with the traditional CP-ABSE approach, which are presented in [57], were coded and designed in a smart contract with the help of the Eth-crypto library [74]. Solidity programming was used to design the smart contract's internal functionality to be deployed on the blockchain simulation platform. The Caliper benchmark tool was used to measure the transaction latency and transaction throughput to determine the schemes' performance [75]. The specification for the simulation setup is presented in Table 7.

Table 7. Simulation Setup.

Network Deployment	**Specifications**
Blockchain network	Ethereum docker container
Consensus protocol	Clique proof-of-authority
Network analyser	Caliper benchmark tool
Smart Contract Deployment	**Specifications**
Programming languages	Vscode Solidity 0.8.0
Cryptographic library	Cryptographic Javascript-functions for the Ethereum (Eth-crypto) library

The experiment was performed in a range of 100–1000 concurrent transactions in different measurements and executed in three rounds of transaction writing to the ledger network. However, in the default network configuration, each round has a range of 50–250 transactions per second (TPS). The smart contracts have been deployed between all the nodes for this set of experiments. The throughput and latency average were calculated

at the end of the third round using the write workload on the basis of the methods invoked in the smart contracts via Equations (30) and (31).

$$\text{Transaction Throughput} = \frac{\text{Total committed transaction}}{\text{Time per second (s)}} \tag{30}$$

$$\begin{aligned}\text{Transaction Latency} = {} & (\text{Transaction execution time} * \text{Network threshold time}) \\ & - \text{Transaction invoke time}\end{aligned} \tag{31}$$

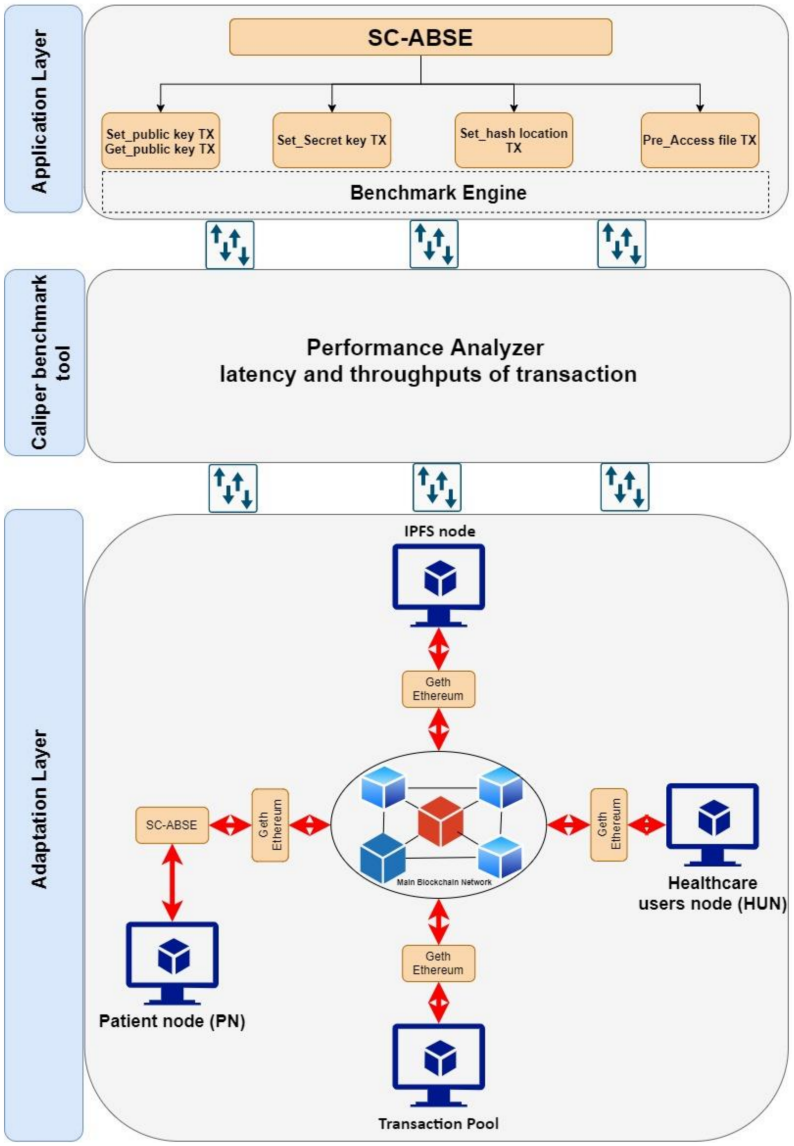


Figure 7. Structure of the blockchain network simulation.

6.2.2. Throughput and Latency Measurements

Transaction throughput: Numerous TPS have been successfully handled by the blockchain network to be included in the block and to be committed as part of the ledger. The throughput was calculated for the proposed scheme SC-ABSE of blockchain-enabled access control for PHRs in the aspect of involvement transactions for system initialisation, SK generation, uploading of health data, and accessing health data. Figure 8 shows the throughput metric comparison between SC-ABSE and the scheme in [57]. The findings show that the SC-ABSE is comparable for all smart contract settings and has a higher throughput of up to an input load of 200 TPS across the entire range of the network.

This can be indicated that SC-ABSE has a good scaling characteristic due to the use of a lightweight cryptographic primitive. Conversely, the scheme in [57] has a fluctuation and a lower rate of throughput transactions of approximately 40 TPS due to the influence of expressive computational operations, as demonstrated in Figure 8.

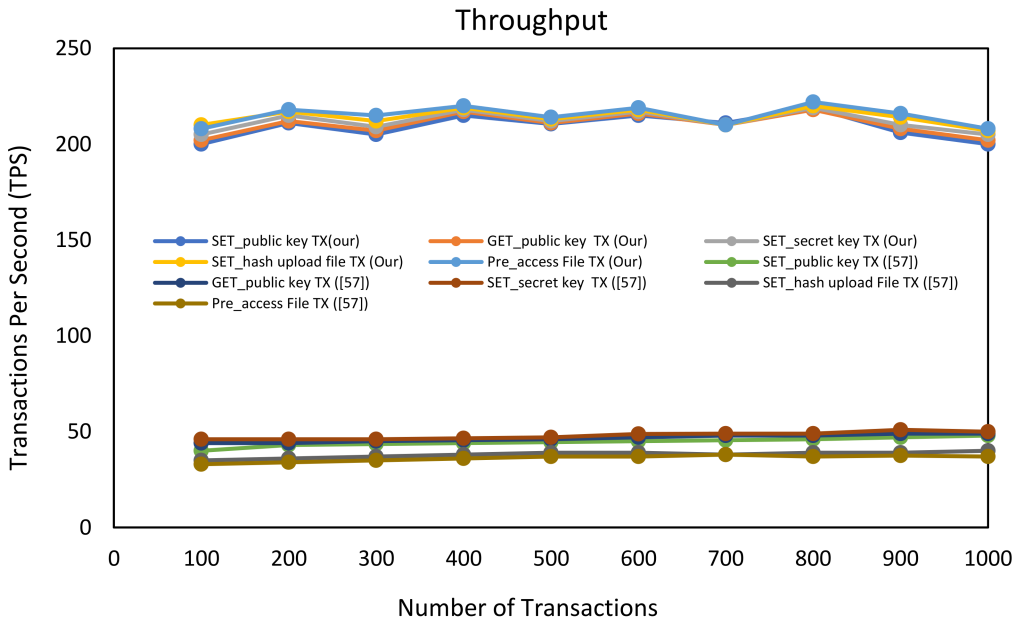


Figure 8. Throughput transaction measurements for smart contract deployments.

Transaction latency: The time elapsed between the request and the confirmation event as received by the users after the transaction is confirmed on the blockchain. Figure 9 shows the average latency measurements for SC-ABSE compared with the scheme in [57]. The access control based on the proposed SC-ABSE scheme in terms of system initialisation for the setting up of public security parameters has been approved to commit the 1000 transactions in an elapsed time of 5.5 s due to the lightweight scheme presented in this study. Simultaneously, the medical data's encryption and storage have tremendously achieved a small scale to conduct 1000 transactions in an elapsed time of 5.9 s. In accessing the health data stored in the IPFS node, the functions used in this transaction are to search, token and decrypt the outsourcing of the data that successfully wrote the transaction in the blockchain network ledger at a rate of 6.9 s of the 1000 transactions committed. This capture results in the simulation of imbalanced times in the transaction when other transactions invoke the smart contract's internal functionality due to outsourcing the medical data stored in the IPFS, but the underlying trend is approximately unaffected. By contrast, the scheme in [57] has a latency measurement of up to 27 s for 1000 transactions as a comparable average for all settings. If the load has been further continued to increase, the latency average tends to start to degrade mainly due to the increased overhead messaging between the nodes and the cryptographic algorithm involved in encrypting and decrypting the data, thus providing additional computational complexity.

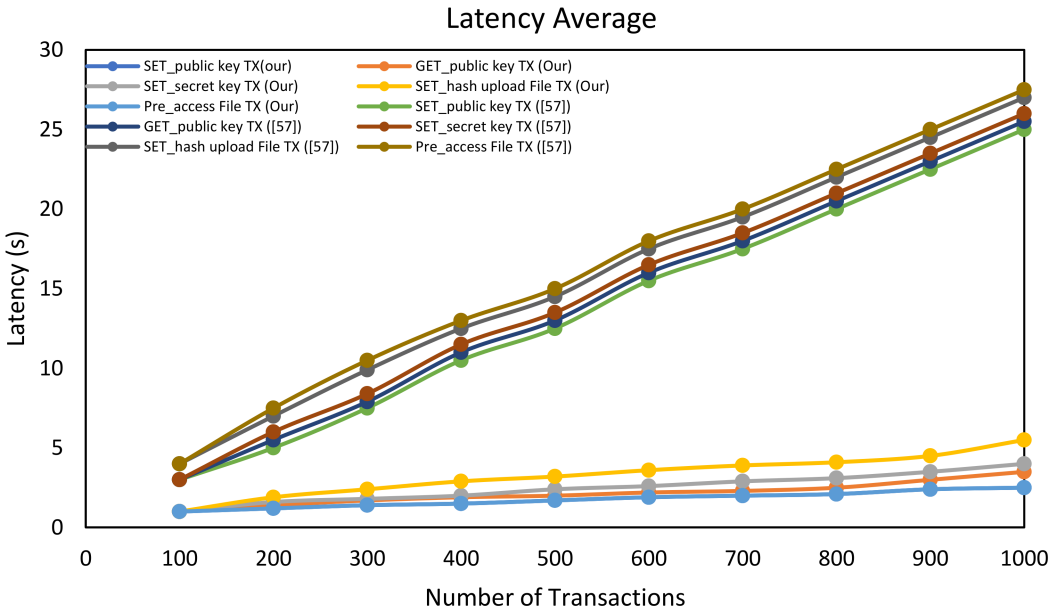


Figure 9. Latency transaction measurements for smart contract deployments.

7. Conclusions and Open Directions

The present study makes several noteworthy contributions to remedy blockchain technology's limitations in terms of privacy and scalability storage over healthcare applications. In addition, this study introduces a novel lightweight cryptographic primitive SC-ABSE to secure outsourced encrypted medical data over IPFS storage. This primitive scheme ensures that the patient-user on the blockchain node controls the search for its outsourced encrypted data under the access control policy without the need to rely on trusted PKGs. Any central authority is eliminated from the proposed scheme, and a single point of failure in the system is prevented. The authorised users in the HUN can outsource the search operations to the IPFS and pressure the IPFS to perform the search accurately. Moreover, the computational operations undertaken in SC-ABSE by the patient-user are exceptionally efficient by leveraging the symmetric encryption algorithm and reducing pairing operations. Moreover, HUN users' consumer is efficiently performed in retrieving medical data due to almost all costly computational operations being offloaded into the IPFS node, and users have been left with extremely lightweight operations. Furthermore, the security definitions and its security resistance of SC-ABSE against the CKA and keyword secrecy are undertaken in the standard model. Moreover, the formal verification tool based on AVISPA proves that the proposed scheme is immune to MIM and replay attacks. The performance analysis measured computational costs, storage costs, communication costs, throughput transactions, and latency transactions from theoretical and practical perspectives. The proposed scheme achieves a high level of security with less computation, storage and communication costs compared with other existing state-of-the-art schemes. The investigation of suggests study in the blockchain network simulation analysis reveals that the throughput that optimises 200 TPS and has a transaction latency is approved to commit 1000 transactions in an elapsed time of 5.5 s. These findings practically imply blockchain's capability and relevance in numerous fields and highlight that blockchain may be the next revolutionary technology to replace the existing healthcare systems.

The generalisability of these results is subject to certain limitations. Firstly, the traditional access control enables authorised users to decrypt the medical data of patients stored in IPFS. However, it hinders first-aid treatment when the patient's life is threatened because on-site first-aid medical personnel cannot obtain the patient's historical medical data. To address this challenge, break-glass access control protocol underneath emergency scenarios is needed in SC-ABSE to allow medical personnel to retrieve the patient's historical medical data stored in IPFS securely and quickly. Secondly, user and attribute revocation mechanisms are needed throughout (BC-ABSE) to revoke or upgrade their attributes in the system. Therefore, future research should concentrate on the limitations described above to meet a proper decentralised healthcare application.

Author Contributions: Conceptualization, H.M.H. and S.M.Y.; Methodology, H.M.H.; Software, H.M.H.; Validation, H.M.H. and S.M.Y.; Formal analysis H.M.H. and S.M.Y.; Investigation, S.M.Y., N.I.U., and M.I.H.N.; Resources, S.M.Y., N.I.U., and M.I.H.N.; Data curation, S.M.Y.; Writing—original draft preparation, H.M.H.; Writing—review and editing, S.M.Y.; Visualisation, H.M.H. and S.M.Y.; Supervision, S.M.Y., N.I.U., and M.I.H.N.; Project administration, H.M.H. and S.M.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work is supported by the Journal Publication Fund of Universiti Putra Malaysia under vote number 9001103. The authors thank Universiti Putra Malaysia for their generous support.

Conflicts of Interest: The authors declare that no conflicts of interest regarding this paper's publication.

Appendix A

The appendix, which contains a full source code of the proposed SC-ABSE scheme, is written in HLPSL codes and validated in the AVISPA.

```
role role_PATIENTNODE (PATIENTNODE, IPFSNODE, HEALTHCAREUSERSNODE:agent,
Skui: symmetric_key,
Kpub:public_key,
H:hash_func,
SND, RCV:channel (dy))
played_by PATIENTNODE
def=
local
State:nat,
Upub:public_key,
R1, Mi, Ni, Gi, Ti, Tta, Yi, Nta, Sj, Tsv, Ys, Tsn, TIDi, Bpj, P, Kpriv,
UIDi, SIDj, DIDi, Message:text,
F:hash_func
init
State := 0
transition
      1. State=0
      /\ RCV (start) =|> State':=1
      /\ Upub' :=new ()
      /\ SND ({Upub' } _Skui)
      2. State=1
      /\RCV ({F (Kpriv.Upub) }_Skui) =|> State':=2
      /\ secret (Kpriv, sec_1, {IPFSNODE})
      /\ R1' :=new ()
      /\ Ti':=new ()
      /\ Mi':=F (R1' .P)
      /\ Ni' :=F (R1' .F (Kpriv.P))
      /\ Gi':=H (H (F (Kpriv.Upub).Ti' .Ni'.UIDi))
      /\ SIDj' :=xor (SIDj, Ni') /\ DIDi' :=xor (UIDi, Ni')
      /\ SND (Gi'.SIDj'.DIDi'.Mi'.Ti')
      /\ witness (PATIENTNODE, IPFSNODE, patient_ipfsnode_ti, Ti')
      /\ witness (PATIENTNODE, IPFSNODE, patient_ipfsnode_r1,R1')
      5. State=2
      /\ RCV (H (SIDj.Sj'.Tta'.TIDi').H(Nta' .F (Kpriv.Upub) .UIDi
      .Tta'.Tsv').xor (xor (TIDi',F (Kpriv.Upub)), Nta').Tsv'.Tta')
      =|> State' :=3
      /\ request (PATIENTNODE, IPFSNODE, patient_ipfsnode_tta, Tta')
      /\ request (PATIENTNODE, IPFSNODE, patient_ipfsnode_tsv, Tsv')
      /\ request (PATIENTNODE, IPFSNODE, patient_ipfsnode_tidi, TIDi')
      /\ Ti' :=new ()
      /\ Yi':=H (H (SIDj.Sj'.Tta'.TIDi'). Ti')
      /\ SND (Yi'.TIDi'.SIDj.Ti')
      7. State=3
      /\ RCV (TIDi.SIDj.xor (Message', H (Bpj'.Tsn')).H (Message'
      . Bpj'. Tsn').Tsn') =|> State':=4
      /\ request (PATIENTNODE, HEALTHCAREUSERSNODE, patient_healthcareusersnode_tsn, Tsn')
end role
```

Figure A1. HLPLS specification of the PN role.

```
role role_IPFSNODE (IPFSNODE, PATIENTNODE, HEALTHCAREUSERSNODE: agent,
Skui: symmetric_key,
Kpub:public_key,
H:hash_func,
SND, RCV:channel (dy))
played_by IPFSNODE
def=
local
State:nat, Upub:public_key, MKj, TTIDi, W, UUpub, TTTIDi,V,
R1, Mi, Ni, Gi, Ti, Tta, Nta, Sj, Tsv, Ys, Tsn, TIDi, Bpj, P, Kpriv,
UIDi, SIDj, DIDi, Message:text,
F:hash_func
init
State := 0
transition
        1. State=0
        /\ RCV ({Upub' }_Skui) =|> State' :=1
        /\ secret (Kpriv, sec_1, {IPFSNODE})
        /\ SND ({F (Kpriv.Upub') }_Skui)
        3. State=1
        /\ RCV (H (F (Kpriv.Upub).Ti'.Ni' .UIDi).xor (SIDj, Ni').xor (UIDi
        ,F(R1' .F (Kpriv.P))) .F (R1'.P).Ti') =|> State' :=2
        /\ request (IPFSNODE, PATIENTNODE, patient_ipfsnode_ti, Ti')
        /\ request (IPFSNODE, PATIENTNODE, patient_ipfsnode_r1, R1')
        /\ secret (Kpriv, sec_1, {IPFSNODE})
        /\TIDi' :=new ()
        /\ Tta' :=new ()
        /\ Sj' := H(SIDj.MKj)
        /\ Tsv' :=new ()
        /\ UUpub':=new()
        /\ TTIDi' :=xor (xor (TTIDi', Sj'), Tta')
        /\ W':=H (SIDj.Sj'.Tta'.Tsv')
        /\ Upub' :=xor (xor (Upub, Sj'), Tta)
        /\ SND (SIDj.TTIDi'.W'.UUpub'.Tsv'.Tta')
        /\ witness (IPFSNODE, HEALTHCAREUSERSNODE, healthcareusersnode_ipfsnode_tta, Tta')
        /\ witness (IPFSNODE, HEALTHCAREUSERSNODE, healthcareusersnode_ipfsnode_tsv, Tsv')
        /\ witness (IPFSNODE, HEALTHCAREUSERSNODE, healthcareusersnode_ipfsnode_tidi, TIDi')
        /\ Nta' := F (Kpriv.F (R1'.P))
        /\ Ys' := H (SIDj.Sj'.Tta'.TIDi')
        /\ V' := H (Nta' .F (Kpriv.Upub).UIDi.Tta'. Tsv')
        /\ TTTIDi' := xor (xor (TIDi',F (Kpriv.Upub)), Nta')
        /\ SND (Ys' .V' .TTTIDi'.Tsv' . Tta')
        /\ witness (IPFSNODE, PATIENTNODE, patient_ipfsnode_tta, Tta')
        /\ witness (IPFSNODE, PATIENTNODE, patient_ipfsnode_tsv, Tsv')
        /\ witness (IPFSNODE, PATIENTNODE, patient_ipfsnode_tidi, TIDi')
        end role
```

Figure A2. HLPLS specification of the IPFS node role.

```
role role_HEALTHCAREUSERSNODE (HEALTHCAREUSERSNODE, PATIENTNODE,
IPFSNODE:agent,
Kpub:public_key,
H:hash_func,
SND, RCV:channel (dy))
played_by HEALTHCAREUSERSNODE
def=
local
State:nat,
Upub:public_key,
TTIDi,Sj,SIDj,Tsn, Z, Tta, Tsv, Ti, Ys, TIDi, Bpj, M, Message: text,
SNjpub:public_key,
Bilinear, Product,F:hash_func
init
State:= 0
transition
        4. State=0
        /\ RCV (SIDj.xor (xor (TTIDi', Sj'), Tta').H (SIDj.Sj'.Tta'
        .Tsv').xor (xor (Upub, Sj'), Tta').Tsv'.Tta') =|> State':= 1
        /\ request (HEALTHCAREUSERSNODE, IPFSNODE, healthcareusersnode_ipfsnode_tta, Tta')
        /\ request (HEALTHCAREUSERSNODE, IPFSNODE, healthcareusersnode_ipfsnode_tsv, Tsv')
        /\ request (HEALTHCAREUSERSNODE, IPFSNODE, healthcareusersnode_ipfsnode_tidi, TIDi)
        6. State=1
        /\ RCV (H (H (SIDj.Sj'.Tta'.TIDi').Ti').TIDi'.SIDj.Ti')
        =|> State':=2
        /\ Bpj' := Bilinear (Upub.Product (SNjpub. Kpub))
        /\ Message' :=new ()
        /\ Tsn':=new ()
        /\ M' := xor(Message' , H (Bpj'. Tsn') )
        /\ Z' :=H (Message'. Bpj'.Tsn')
        /\ SND (TIDi'.SIDj.M'.Z' .Tsn')
        /\ witness (HEALTHCAREUSERSNODE, PATIENTNODE, patient_healthcareusersnode_tsn,
Tsn')
        end role
```

Figure A3. HLPLS specification of the HUN role.

```
role session (PATIENTNODE, IPFSNODE, HEALTHCAREUSERSNODE: agent,
Skui: symmetric_key,
Kpub:public_key,
H: hash_func)
def=
local
SND3, RCV3, SND2, RCV2, SND1, RCV1:channel (dy)
composition
        role_HEALTHCAREUSERSNODE (HEALTHCAREUSERSNODE, PATIENTNODE, IPFSNODE,
Kpub, H, SND3, RCV3)
        /\ role_IPFSNODE (IPFSNODE, PATIENTNODE, HEALTHCAREUSERSNODE, Skui, Kpub, H,
SND2, RCV2)
        /\ role_PATIENTNODE (PATIENTNODE, IPFSNODE, HEALTHCAREUSERSNODE, Skui,
Kpub, H, SND1, RCV1)
        end role
```

Figure A4. HLPLS specification of the session role.

```
role environment ()
def=
const
patientnode, ipfsnode, healthcareusersnode:agent, tsn,tta, ti, const_1: text,
skui: symmetric_key, kpub:public_key, h:hash_func,
sec_1, sec_2, sec_3, patient_ipfsnode_tta, patient_ipfsnode_r1, patient_ipfsnode_tsv, pa-
tient_ipfsnode_tidi, patient_ipfsnode_ti, healthcareusersnode_ipfsnode_tta,
healthcareusersnode_ipfsnode_tsv, healthcareusersnode_ipfsnode_tidi, pa-
tient_healthcareusersnode_tsn:protocol_id
intruder_knowledge = {h, ti,tta, tsn}
composition
        session (patientnode,ipfsnode, healthcareusersnode, skui, kpub, h)
        /\ session(i, ipfsnode, healthcareusersnode, skui, kpub, h)
        /\session (patientnode, i, healthcareusersnode, skui, kpub, h)
        /\session (patientnode, ipfsnode, i, skui, kpub, h)
end role
        goal
        secrecy_of sec_1, sec_2, sec_3
        authentication_on patient_ipfsnode_tta
        authentication_on patient_ipfsnode_tsv
        authentication_on patient_ipfsnode_tidi
        authentication_on healthcareusersnode_ipfsnode_tta
        authentication_on healthcareusersnode_ipfsnode_tsv
        authentication_on healthcareusersnode_ipfsnode_tidi
        authentication_on patient_healthcareusersnode_tsn
        authentication_on patient_ipfsnode_ti
        authentication_on patient_ipfsnode_r1
        end goal
        environment ()
```

Figure A5. HLPLS specification of the environment role.

References

1. Khezr, S.; Rachid, B.; Abdulsalam, Y. Blockchain-based Model for Sharing Activities of Daily Living in Healthcare Applications. In Proceedings of the 2020 IEEE Int. Conf. on Dependable, Autonomic and Secure Computing, Int. Conf. on Pervasive Intelligence and Computing, Calgary, AB, Canada, 17–22 August 2020; pp. 627–633.
2. Justinia, T. Blockchain Technologies: Opportunities for Solving Real-World Problems in Healthcare and Biomedical Sciences. *Acta Inform. Med.* **2019**, *27*, 284. [CrossRef]
3. Gul, M.J.; Subramanian, B.; Paul, A.; Kim, J. Blockchain for public health care in smart society. *Microprocess. Microsyst.* **2021**, *80*, 103524. [CrossRef]
4. Mayer, A.H.; da Costa, C.A.; Righi, R.D.R. Electronic health records in a blockchain: A systematic review. *Health Inform. J.* **2020**, *26*, 1273–1288. [CrossRef]
5. Hasselgren, A.; Kralevska, K.; Gligoroski, D.; Pedersen, S.A.; Faxvaag, A. Blockchain in healthcare and health sciences—A scoping review. *Int. J. Med. Inform.* **2020**, *134*, 104040. [CrossRef]
6. Hussien, H.M.; Yasin, S.M.; Udzir, S.N.I.; Zaidan, A.A.; Zaidan, B.B. A systematic review for enabling of develop a blockchain technology in healthcare application: Taxonomy, substantially analysis, motivations, challenges, recommendations and future direction. *J. Med. Syst.* **2019**, *43*, 320. [CrossRef]
7. Mazlan, A.A.; Daud, S.M.; Sam, S.M.; Abas, H.; Rasid, S.Z.A.; Yusof, M.F. Scalability Challenges in Healthcare Blockchain System—A Systematic Review. *IEEE Access* **2020**, *8*, 23663–23673. [CrossRef]
8. Xu, J.; Xue, K.; Li, S.; Tian, H.; Hong, J.; Hong, P.; Yu, N. Healthchain: A blockchain-based privacy preserving scheme for large-scale health data. *IEEE Internet Things J.* **2019**, *6*, 8770–8781. [CrossRef]
9. Nizamuddin, N.; Salah, K.; Azad, M.A.; Arshad, J.; Rehman, M.H. Decentralized document version control using ethereum blockchain and IPFS. *Comput. Electr. Eng.* **2019**, *76*, 183–197. [CrossRef]
10. Nguyen, D.C.; Pathirana, P.N.; Ding, M.; Seneviratne, A. Blockchain for secure ehrs sharing of mobile cloud based e-health systems. *IEEE Access* **2019**, *7*, 66792–66806. [CrossRef]
11. Pournaghi, S.M.; Bayat, M.; Farjami, Y. MedSBA: A novel and secure scheme to share medical data based on blockchain technology and attribute-based encryption. *J. Ambient Intell. Humaniz. Comput.* **2020**, *11*, 4613–4641. [CrossRef]

12. Iqbal, J.; Umar, A.I.; Amin, N.; Waheed, A. Efficient and secure attribute-based heterogeneous online/offline signcryption for body sensor networks based on blockchain. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719875654. [CrossRef]
13. Wang, H.; Song, Y. Secure cloud-based EHR system using attribute-based cryptosystem and blockchain. *J. Med. Syst.* **2018**, *42*, 152. [CrossRef]
14. Thwin, T.T.; Vasupongayya, S. Blockchain-based access control model to preserve privacy for personal health record systems. *Secur. Commun. Netw.* **2019**. [CrossRef]
15. Khatoon, A. A blockchain-based smart contract system for healthcare management. *Electronics* **2020**, *9*, 94. [CrossRef]
16. Xia, Q.I.; Sifah, E.B.; Asamoah, K.O.; Gao, J.; Du, X.; Guizani, M. MeDShare: Trust-less medical data sharing among cloud service providers via blockchain. *IEEE Access* **2017**, *5*, 14757–14767. [CrossRef]
17. Fan, K.; Wang, S.; Ren, Y.; Li, H.; Yang, Y. Medblock: Efficient and secure medical data sharing via blockchain. *J. Med. Syst.* **2018**, *42*, 136. [CrossRef] [PubMed]
18. Zhao, Y.; Cui, M.; Zheng, L.; Zhang, R.; Meng, L.; Gao, D.; Zhang, Y. Research on electronic medical record access control based on blockchain. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719889330. [CrossRef]
19. Abouelmehdi, K.; Beni-Hessane, A.; Khaloufi, H. Big healthcare data: Preserving security and privacy. *J. Big Data* **2018**, *5*, 1. [CrossRef]
20. Hathaliya, J.J.; Tanwar, S. An exhaustive survey on security and privacy issues in Healthcare 4.0. *Comput. Commun.* **2020**, *153*, 311–335. [CrossRef]
21. Song, D.X.; Wagner, D.; Perrig, A. Practical techniques for searches on encrypted data. In Proceedings of the 2000 IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 14–17 May 2000; pp. 44–55.
22. Curtmola, R.; Garay, J.; Kamara, S.; Ostrovsky, R. Searchable symmetric encryption: Improved definitions and efficient constructions. *J. Comput. Secur.* **2011**, *19*, 895–934. [CrossRef]
23. Wang, C.; Cao, N.; Li, J.; Ren, K.; Lou, W. Secure ranked keyword search over encrypted cloud data. In Proceedings of the IEEE 30th International Conference on Distributed Computing Systems, Genoa, Italy, 21–25 June 2010; pp. 253–262.
24. Cao, N.; Wang, C.; Li, M.; Ren, K.; Lou, W. Privacy-preserving multi-keyword ranked search over encrypted cloud data. *IEEE Trans. parallel Distrib. Syst.* **2013**, *25*, 222–233. [CrossRef]
25. Alderman, J.; Martin, K.M.; Renwick, S.L. Multi-level access in searchable symmetric encryption. In Proceedings of the International Conference on Financial Cryptography and Data Security, Sliema, Malta, 3–7 April 2017; Springer: Cham, Switzerland, 2017; pp. 35–52.
26. Liu, C.; Zhu, L.; Chen, J. Efficient searchable symmetric encryption for storing multiple source dynamic social data on cloud. *J. Netw. Comput. Appl.* **2017**, *86*, 3–14. [CrossRef]
27. Emura, K.; Ito, K.; Ohigashi, T. Secure-channel free searchable encryption with multiple keywords: A generic construction, an instantiation, and its implementation. *J. Comput. Syst. Sci.* **2020**, *114*, 107–125. [CrossRef]
28. Chen, L.; Zhang, N.; Sun, H.M.; Chang, C.C.; Yu, S.; Choo, K.K.R. Secure search for encrypted personal health records from big data NoSQL databases in cloud. *Computing* **2020**, *102*, 1521–1545. [CrossRef]
29. Mihailescu, M.I.; Nita, S.L.; Racuciu, C. Multi-level access using searchable symmetric encryption with applicability for earth sciences. Scientific Bulletin "Mircea cel Batran". *Nav. Acad.* **2020**, *23*, 213A–220A.
30. Li, H.; Zhang, F.; He, J.; Tian, H. A searchable symmetric encryption scheme using blockchain. *arXiv* **2017**, arXiv:1711.01030.
31. Li, H.; Tian, H.; Zhang, F.; He, J. Blockchain-based searchable symmetric encryption scheme. *Comput. Electr. Eng.* **2019**, *73*, 32–45. [CrossRef]
32. Zhang, Y.; Deng, R.H.; Shu, J.; Yang, K.; Zheng, D. TKSE: Trustworthy keyword search over encrypted data with two-side verifiability via blockchain. *IEEE Access* **2018**, *6*, 31077–31087. [CrossRef]
33. Jiang, P.; Guo, F.; Liang, K.; Lai, J.; Wen, Q. Searchain: Blockchain-based private keyword search in decentralised storage. *Future Gener. Comput. Syst.* **2020**, *107*, 781–792. [CrossRef]
34. Chen, L.; Lee, W.K.; Chang, C.C.; Choo, K.K.R.; Zhang, N. Blockchain based searchable encryption for electronic health record sharing. *Future Gener. Comput. Syst.* **2019**, *95*, 420–429. [CrossRef]
35. Chen, Y.; Ding, S.; Xu, Z.; Zheng, H.; Yang, S. Blockchain-based medical records secure storage and medical service framework. *J. Med. Syst.* **2019**, *43*, 5. [CrossRef] [PubMed]
36. Cao, S.; Zhang, G.; Liu, P.; Zhang, X.; Neri, F. Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain. *Inf. Sci.* **2019**, *485*, 427–440. [CrossRef]
37. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 30 October 2006; pp. 89–98.
38. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
39. Lewko, A.; Okamoto, T.; Sahai, A.; Takashima, K.; Waters, B. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 62–91.
40. Balu, A.; Kuppusamy, K. An expressive and provably secure ciphertext-policy attribute-based encryption. *Inf. Sci.* **2014**, *276*, 354–362. [CrossRef]

41. Pirretti, M.; Traynor, P.; McDaniel, P.; Waters, B. Secure attribute-based systems. *J. Comput. Secur.* **2010**, *18*, 799–837. [CrossRef]
42. Nita, S.L.; Mihailescu, M.I. A Searchable Encryption Scheme Based on Elliptic Curves. In Proceedings of the Workshops of the International Conference on Advanced Information Networking and Applications, Caserta, Italy, 15–17 April 2020; Springer: Cham, Switzerland, 2020; pp. 810–821.
43. Hur, J.; Noh, D.K. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Trans. Parallel Distrib. Syst.* **2010**, *22*, 1214–1221. [CrossRef]
44. Yin, H.; Zhang, J.; Xiong, Y.; Ou, L.; Li, F.; Liao, S.; Li, K. CP-ABSE: A ciphertext-policy attribute-based searchable encryption scheme. *IEEE Access* **2019**, *7*, 5682–5694. [CrossRef]
45. Sun, J.; Ren, L.; Wang, S.; Yao, X. Multi-keyword searchable and data verifiable attribute-based encryption scheme for cloud storage. *IEEE Access* **2019**, *7*, 66655–66667. [CrossRef]
46. Wang, S.; Yao, L.; Chen, J.; Zhang, Y. KS-ABESwET: A Keyword Searchable Attribute-Based Encryption Scheme with Equality Test in the Internet of Things. *IEEE Access* **2019**, *7*, 80675–80696. [CrossRef]
47. Sultan, N.H.; Kaaniche, N.; Laurent, M.; Barbhuiya, F.A. Authorised keyword search over outsourced encrypted data in cloud environment. *IEEE Trans. Cloud Comput.* **2019**. early access. [CrossRef]
48. Zhang, L.; Hu, G.; Mu, Y.; Rezaeibagha, F. Hidden ciphertext policy attribute-based encryption with fast decryption for personal health record system. *IEEE Access* **2019**, *7*, 33202–33213. [CrossRef]
49. Li, M.; Yu, S.; Ren, K.; Lou, W. September. Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings. In *International Conference on Security and Privacy in Communication Systems*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 89–106.
50. Li, M.; Yu, S.; Zheng, Y.; Ren, K.; Lou, W. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Trans. parallel Distrib. Syst.* **2012**, *24*, 131–143. [CrossRef]
51. Prince, P.B.; Lovesum, S.J. Privacy Enforced Access Control Model for Secured Data Handling in Cloud-Based Pervasive Health Care System. *SN Comput. Sci.* **2020**, *1*, 1–8. [CrossRef]
52. Xu, L.; Xu, C.; Liu, J.K.; Zuo, C.; Zhang, P. Building a dynamic searchable encrypted medical database for multi-client. *Inf. Sci.* **2020**, *527*, 394–405. [CrossRef]
53. Guo, R.; Shi, H.; Zheng, D.; Jing, C.; Zhuang, C.; Wang, Z. Flexible and efficient blockchain-based ABE scheme with multi-authority for medical on demand in telemedicine system. *IEEE Access* **2019**, *7*, 88012–88025. [CrossRef]
54. Liu, X.; Ma, W.; Cao, H. MBPA: A Medibchain-Based Privacy-Preserving Mutual Authentication in TMIS for Mobile Medical Cloud Architecture. *IEEE Access* **2019**, *7*, 149282–149298. [CrossRef]
55. Wang, S.; Zhang, D.; Zhang, Y. Blockchain-based personal health records sharing scheme with data integrity verifiable. *IEEE Access* **2019**, *7*, 102887–102901. [CrossRef]
56. Niu, S.; Chen, L.; Wang, J.; Yu, F. Electronic Health Record Sharing Scheme With Searchable Attribute-Based Encryption on Blockchain. *IEEE Access* **2019**, *8*, 7195–7204. [CrossRef]
57. Sun, J.; Yao, X.; Wang, S.; Wu, Y. Blockchain-Based Secure Storage and Access Scheme For Electronic Medical Records in IPFS. *IEEE Access* **2020**, *8*, 59389–59401. [CrossRef]
58. Wang, S.; Wang, X.; Zhang, Y. A secure cloud storage framework with access control based on blockchain. *IEEE Access* **2019**, *7*, 112713–112725. [CrossRef]
59. Miao, Y.; Ma, J.; Liu, X.; Wei, F.; Liu, Z.; Wang, X.A. m 2-ABKS: Attribute-based multi-keyword search over encrypted personal health records in multi-owner setting. *J. Med. Syst.* **2016**, *40*, 246. [CrossRef]
60. Zheng, Q.; Xu, S.; Ateniese, G.A. VABKS: Verifiable attribute-based keyword search over outsourced encrypted data. In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014; pp. 522–530.
61. Ali, M.; Sadeghi, M.R. Provable secure lightweight attribute-based keyword search for cloud-based Internet of Things networks. *Trans. Emerg. Telecommun. Technol.* **2020**, e3905. [CrossRef]
62. The AVISPA Project. Available online: http://www.avispa-project.org/ (accessed on 2 February 2021).
63. Armando, A.; Basin, D.; Boichut, Y.; Chevalier, Y.; Compagna, L.; Cuéllar, J.; Drielsma, P.H.; Héam, P.C.; Kouchnarenko, O.; Mantovani, J.; et al. The AVISPA tool for the automated validation of internet security protocols and applications. In *International Conference on Computer Aided Verification*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 281–285.
64. Chatterjee, S.; Das, A.K. An effective ECC-based user access control scheme with attribute-based encryption for wireless sensor networks. *Secur. Commun. Netw.* **2015**, *8*, 1752–1771. [CrossRef]
65. Ding, S.; Cao, J.; Li, C.; Fan, K.; Li, H. A novel attribute-based access control scheme using blockchain for IoT. *IEEE Access* **2019**, *7*, 38431–38441. [CrossRef]
66. Hsu, C.L.; Chen, W.X.; Le, T.V. An Autonomous Log Storage Management Protocol with Blockchain Mechanism and Access Control for the Internet of Things. *Sensors* **2020**, *20*, 6471. [CrossRef] [PubMed]
67. Kim, M.; Yu, S.; Lee, J.; Park, Y.; Park, Y. Design of secure protocol for cloud-assisted electronic health record system using blockchain. *Sensors* **2020**, *20*, 2913. [CrossRef] [PubMed]
68. SPAN: Security Protocol ANimator for AVISPA. Available online: http://people.irisa.fr/Thomas.Genet/span/ (accessed on 2 February 2021).
69. Enron Email Dataset. Available online: https://www.cs.cmu.edu/~{}enron/ (accessed on 11 November 2020).

70. Type-A Pairings-PBC. Available online: https://crypto.stanford.edu/pbc/manual/ch08s03.html (accessed on 11 November 2020).
71. Pairing Based Cryptography (PBC) Library. Available online: https://crypto.stanford.edu/pbc/ (accessed on 11 November 2020).
72. Ethereum Blockchain Network. Available online: https://geth.ethereum.org/getting-started/private-net (accessed on 11 November 2020).
73. Clique Proof-of-Authority Consensus Protocol. Available online: https://github.com/ethereum/go-ethereum/tree/master/consensus/clique (accessed on 11 November 2020).
74. Eth-Crypto Library for Ethereum. Available online: https://github.com/pubkey/eth-crypto (accessed on 11 November 2020).
75. Caliper Benchmark Tool for Ethereum Configuration. Available online: https://hyperledger.github.io/caliper/v0.4.2/ethereum-config/ (accessed on 11 November 2020).