Healthcare Applications Using Blockchain with a Cloud-Assisted Decentralized Privacy-Preserving Framework

B D Deebak, Seong Oun Hwang*, *Senior Member, IEEE*

***Abstract*—In recent times, cloud-enabled healthcare services have gained much attention in fulfilling the analysis of privacy risks associated with effective decision management systems including trustworthiness and secure data sharing. This system has revolutionized the medical architecture to evolve unprecedented opportunities in using the technologies of next-generation networks, including services, control, and signaling, to effectively improve content delivery to individuals and organizations. However, it is a highly complex matter to distribute confidential data over a public network because data privacy and device security are at risk. As a result, a cloud-based healthcare application is introduced that integrates the computation capabilities of ubiquitous computing to provide extensive communications over dedicated Internet access in order to store health records. To manage access control mechanisms and process sensitive data without extensive computation, the existing cloud-centric systems access remote servers via dedicated networks. Based on a centralized architecture, a dedicated network uses different application domains to deliver information to the healthcare industry. Unfortunately, computation complexity greatly deteriorates the performance of peer-to-peer (P2P) communications. Thus, this paper presents a cloud-assisted decentralized privacy preserving framework (CA-DPPF) using blockchain and key agreement (KA) mechanisms to achieve secure data storage and privacy. The detailed security analysis proves that the proposed scheme fulfills the desired security properties of healthcare supply chain management (H-SCM) such as conditional traceability, data immutability, and data integrity. Overall, exploratory analysis shows that CA-DPPF guarantees better transaction efficiencies such as less latency and more throughput in order to improve the service utilization factor.**

***Index Terms*—Internet of Things, Cloud, Peer-to-Peer Communication, Privacy, Blockchain, Decentralized Systems, Service Utilization**

I. Introduction

Of late, various electronic healthcare (e-Healthcare) applications like EHR have emerged in clinical laboratories and pharmacies for digital-centric orchestration enabling a role of supply chain management. The concept of healthcare supply chain management (H-SCM) deals with production, goods delivery, data transactions, financial payments, and overseas transfers as the backbone of commerce and to reduce costs. It eases industry functions for effectively managing the supply chain [1] and determining the existence of fake products. In the supply chain process, rigorous work is being initiated to identify types of products. Typically, H-SCM applies electronic tags, including barcodes and radio frequency identification (RFID) tags, to maintain effective service exchanges. Therefore, the H-SCM system is challenged in protecting the transparency and privacy of commercial products over wireless channels.

Most importantly, commercial products cannot resist well-known intrusions such as replay and masquerade attacks, URL redirection, and eavesdropping[2]. For private data access, a proper authentication mechanism may be preferable for preventing data forgery in the supply chain. It can provide better authentication between systems and real-time entities to ensure secure channel access. Blockchain is emerging as a key technology to enhance digitization in healthcare sectors. It transforms traditional medical systems into a digitized healthcare ecosystem to decentralize the process of medical assistance. It uses information and communication technology (ICT) to provide modern digitization between patients and service providers [2]. The applications of blockchain technology provide an interactive platform to manage medical records, treatment processes, payment claims, medical data diagnoses, data verification, auditing, and dispensary assistance.

A few representative information technologies, such as cyber physical systems, cloud/edge computing, and Internet of Medical Things (IoMT) design the consensus operation in accordance with the phases of the blockchain ledger including endorsement, ordering, and validation to maintain trustworthiness in a non-trusted environment [3]. Moreover, the IoMT uses peer-to-peer (P2P) transactions to develop various applications that enable faster transactions and greater trustworthiness, eliminating issues with scalability and processing overhead. It may connect a huge volume of physical objects with distributed modular architecture like blockchain to collect and retrieve environmental contents that obtain its privacy preservation with hash encryption like SHA-256 in order to authorize data storage. In particular, a P2P network limits two significant factors (transmission rate and reliability) to disseminate network transactions [4]. Such a network can ensure a fair transaction process to change unreliable network connections considerably.

According to experimental insights, the Ethereum network has an average latency of six seconds and 10 seconds, respectively, in creating a data block received by 50% and 90% of the overall nodes [5]. This network latency may degrade blockchain performance in terms of transactions per second, which leads to the potential risk of generating a fork.

A. Research Motivation

Most industrial applications like banking and healthcare are progressing towards the transformation of a blockchain-based distributed environment by virtue of its distributed ledger, enhanced security, immutable, and auditable. In practice, this environment provides an influential distributed ledger technology to specify the keystones of the data storage mechanism which creates an irreversible ledger to facilitate the participation of the communication

*Corresponding Authors (Seong Oun Hwang)

This work was supported by the Brain Pool Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and Information Communication Technology (ICT) under Grant No. 2022H1D3A2A02081848 and by the Gachon University research fund under Grant GCU-202206210001.

B.D. Deebak and Seong Oun Hwang are with the Department of Computer Engineering, Gachon University, Seongnam 13120, South Korea, (Email: deebak@gachon.ac.kr, sohwang@gachon.ac.kr).

entities in the blockchain network. As a consequence, the maturity level of the blockchain (i.e., 3.0) deals with the interface of the decentralized application (dApp) to provide a few key innovative mechanisms including key indexing and query processing. The primary goal of the dApp is to enhance the range of application services that include either physical or virtual objects to retrieve data contents, regardless of their purpose. However, this interface is only accessible to authorized users. Otherwise, it might experience security risks including data theft, fabrication, and violation.

Omar et al. [6] considered a tool of the Internet of Medical Things (IoMT) to relate healthcare information with a distributed peer-to-peer network (P2P) using consortium blockchain as a system of systems realizing multiple use-case scenarios to examine user services. However, due to resource limitations, and device heterogeneity, the existing solutions cannot be adopted in eco-friendly environments. Moreover, security technologies may incur additional computation costs to provide better performance efficiency, which is necessary for a centralized architecture. There may be some scalability issues while various security approaches are proposed to integrate novel scenarios and services [7].

In other words, a blockchain-based solution uses a specific node in the network so-called miner to secure a high stake or assets which employ users' public keys to create a transaction whereby a high level of anonymity is introduced. Thus, the computing peers of the H-SCM involve distributed storage to maintain the healthcare information as it is immutable by its nature to protect the network channels in order to share the transaction details with the connected peers. Taking this benefit into account, the proposed CA-DPPF uses a secure key agreement and encryption mechanism to authenticate the identities of the H-SCM i.e., enterprise networks [8]. The encryption mechanism applies an elliptic curve digital signature algorithm (ECDSA) in the procedural process of key agreement to access and retrieve the electronic healthcare record (EHR) processed by the computing peers.

B. Research Contribution

Most intruders search for an insecure channel and determine the network or cloud service provider in order to gain system access. They may exploit sensitive information to cause severe damage to network providers.One recent study revealed that many service providers may be in a state of data loss serious enough to degrade system efficiencies, including performance and security [9]. Most of the sensitive information is stored in the cloud without a proper encryption mechanism to address the issue of data privacy, allowing an intruder to easily steal private patient information. A typical scenario reveals that medical data should be preserved in an EHR system to maintain flow efficiencies.

Therefore, this paper contributes a few key technical strategies such as privacy-preserving, system logs, access control, and storage to manage sensitive information. The major contributions are as follows:

1) We present a cloud-assisted decentralized privacy preserving framework (CA-DPPF) using blockchain to achieve secure data transmission, storage, and transfer with privacy protection using KA based on an elliptic-curve digital signature algorithm (EC-DSA).
2) We apply an ECC-based digital signature algorithm with forward security to preserve the privacy of the data in the cloud-assisted decentralized networks which operate double parameters in the signature and verification process to prevent insecure or weak randomness attack whereby storage efficiency is guaranteed while generating the cryptographic keys.
3) We design a consensus mechanism of proof of trusted authority (POTA) which protects medical data in cloud storage using a voter-per-share policy in order to handle an efficient flow for data transmission. Moreover, this design strategy provides vote casting to the stakeholders not only to maintain better consistency with the Ethereum network but also to evaluate the credibility of the transaction like a proper signature verification process.
4) We develop a new peer-to-peer registration process using the KA mechanism to verify the legitimacy of the client registration request. In addition, this registration process utilizes a strategy of system log transactions using POTA to preserve data integrity and access control of the enterprise network, which allows the clients to hide their identities while performing an encryption/decryption process to ensure user anonymity and preserve privacy.
5) Finally, we build a real-time setup using Hyperledger Fabric to view the execution process of the transactions via consortium blockchain and analyze the performance of the communication metrics such as committed transactions, average latency, and throughput over the proposed CA-DPPF and other existing frameworks.

The remaining sections of the paper are as follows. Section II addresses the major challenges for blockchain technology, as found in related work. Section III introduces system preliminaries that include consensus mechanisms, Hyperledger Fabric, and a system model to signify the use of a cloud-centric framework with privacy preservation. Section IV discusses the proposed CA-DPPF and its algorithms, smart contracts, blockchain, system security, and KA based on EC-DSA. Section V and VI present the formal, informal, cost efficiency, and experimental analysis to prove the significance of the transaction process in terms of security and relevant quality metrics. Finally, Section VII summarizes this research.

II. Related Works

Of late, innovators and scientific researchers have addressed a few of the technical benefits of blockchain technology, such as consensus algorithms and cryptographic techniques that show the salient features of distributed ledger technology at different levels of standardization [10]. Moreover, business standards such as ISO, FPL, and ISDA cannot be associated with a member of a blockchain network to inspect key features such as decentralization, transparency, and traceability [11]. In line with the distribution of ledger technology, reference data cannot be tampered with in order to guarantee secure storage and trustworthy computations. Of late, various authentication schemes have been proposed for electronic healthcare systems [12]. Most of the existing schemes ensure security efficiencies in EHR systems that integrate a cloud-based environment in order to improve the efficiency rate of the systems in terms of computation, communication, and storage costs. Kaur et al. [13] and Nagasubramanian et al. [14] integrated key features of disruptive technologies, such as sensors, communications, blockchains, and cloud computing, to analyze the degree of a decentralization framework. However, their schemes do not handle the computation phase effectively enough to analyze system log transactions.

Cai et al. [23] summarized various computing techniques in access control, including attribute-based, encryption-based, and task-based control. These techniques were further compared with a few more key metrics (namely, confidentiality, authorization, expansibility, and security) to identify the security levels of the virtual servers and the cloud platform. Rouhani and Deters [24] investigated the built-in aspects of access control to handle technical issues in blockchain technology. Also, a few existing solutions were classified to help understand the systematic approaches of transactions, smart contracts, data sharing, and cloud federations. Esposito et al. [15] designed an effective solution for distributed identity management and authorization policies in order to analyze a single sign-on mechanism among other networking architectures. This technique integrates an open-source technology known as FIWARE to examine the security policies of smart cities.

Khalid et al. [16] presented a decentralized access control system for lightweight IoT applications, which utilizes the properties of fog computing, blockchain technology, and peer-to-peer connectivity to secure device connectivity. Patwary et al. [17] developed a decentralized framework using blockchain to authenticate IoT devices independently without trusted third parties. This framework applies an Ethereum smart contract to inspect the property of mutual authentication between fog computing devices. Three-factor authentication was designed using ECC for smart healthcare [25]. This authentication scheme prevents most of the potential attacks, including replay, privileged insider, and forgery. Most importantly, it could not solve the single point of failure problem because it has a centralized system to gain data ownership. Pandey and Litoriya [26] proposed a security framework to analyze medical counterfeiting by using blockchain. Their scheme achieves vital features such as data integrity and security to enhance the privacy rate of the application systems. Agbo and Mahmoud [27] presented a comparative framework for electronic healthcare systems using blockchain. Tanwar et al. [28] designed a blockchain-based security framework to examine medical data. However, their scheme could not resolve the issue of system reliability in analyzing trusted rates of third parties.

Wang et al. [18] introduced a cloud-assisted framework to improve the security and privacy of a system using blockchain. Their scheme applies searchable and proxy re-encryption to signify the use of control access and security. Chen et al. [29] presented a secure storage system to manage blockchain by using cloud computing techniques. Their system uses bilinear pairing to mitigate the computation and storage costs. Karumba et al. [19] designed a hypergraph-based adaptive framework to address key issues of peer-to-peer energy trading systems, such as scalability and privacy. It uses data tagging and an anonymization model to specify attribute transactions of network clusters. Samir et al. [20] introduced a trustworthy decentralized framework to integrate a secret share scheme with blockchain-based smart contract technologies. This framework applies digital identity management and self-sovereign identity to preserve the identity credentials of IoT devices. However, the decentralized framework fails to contribute an anonymization model between the network clusters to achieve better data sharing in order to improve the efficiency rates of EHR systems i.e., computing resources.

Gao et al. [30] presented a privacy preserving authentication using an elliptic curve digital signature algorithm to protect the real-time information and to minimize the storage overhead caused by massive device connectivity. Liu et al. [31] designed a blockchain-based anonymous authentication using an elliptic-curve encryption system to store and access computing data via a blockchain network. Their mechanism uses a chameleon hash function to generate a few quantities of authentication elements to minimize the cost efficiency of the mutual authentication phase. Zhang et al. [21] constructed a privacy preserving blockchain using multi-factor key derivation to prevent the potential loss of storage cost. This scheme uses a hardware-assisted key derivation to verify the unlinkable identities of the computing devices. Zhang et al. [22] addressed the issue of anti-single-point failure using conditional privacy preserving authentication framework for vehicular ad hoc networks (VANET). This framework realizes dynamic revocation through a smart contract to track the activities of illegal vehicles. However, in the performance evaluation, the authors did not consider any realistic network like hyper-ledger fabric or distributed database to analyze the challenges of peer-to-peer networks (i.e., decentralization and immutability).

Roy et al. [32] presented a fine-grained access control over the cloud to secure multi-server data with provable authentication. Additionally, in this approach, the computing phases including setup and registration are not involved with the registration center to protect the server against eavesdropping attacks. Unfortunately, this access control mechanism fails to prove its significance in multiple authentication methods to verify client identities. Srinivas et al. [33] developed a cloud-based user authentication framework to establish secure communication among the users and wearable sensing units. This authentication framework applies the current system timestamp with registration and revocation phases to protect communication against replay attacks. Regardless, in the security analysis, this framework pretermitted a few attacks including denial of service and man-in-the-middle to validate its resiliency in cloud-centric applications. Wazid et al. [34] designed a blockchain-enabled security mechanism to prevent ransomware attacks in smart healthcare applications. Though, this model does not include a formal security analysis like the real-or-random (RoR) model to analyze the quality of the parameters deployed in this mechanism to protect the integrity and confidentiality of the data.

In recent times, researchers have employed decentralization technology and blockchain network to offer a promising solution that solves the problem of centralization and allows users to access the system with a similar identity in various application scenarios [35], [36]. However, the promising solution cannot manage the voluminous access without any service providers to reduce the cost efficiency and meet cloud computing systems' privacy protection. Therefore, in this paper, we present the cloud-assisted decentralized privacy preserving framework using blockchain and EC-DSA, which exploits a key strategy of system log transactions, signature, and verification process to preserve the identities of the computing devices. Table I shows a summary of related work on blockchain and cloud-assisted frameworks for healthcare applications.

III. System Preliminaries

This section introduces the basic system preliminaries including the digital signature algorithm, consensus mechanisms, Hyperledger Fabric, and system model to elaborate on service decentralization.

TABLE I: Brief Summaries of Related Works on Blockchain and Cloud-Assisted Framework for Healthcare Applications

Previous Works	Solution Description	Applied Framework	Limitation
Esposito et al. [15]	Developing a distributed identity management system and authorization policies to examine the single sign-on mechanism.	Blockchain-Based Access Control.	• Vulnerable to distributed denial of service attacks as the proposed scheme does not have any specific strategy to deal with decentralized processing architecture in order to eliminate the chances of single point of failure. • Does not have any applied strategy of service utilization to verify data integrity and user privacy.
Khalid et al. [16]	Designing a decentralized access control system to exploit the key properties of fog computing and blockchain technology.	Distributed Authentication and Authorization Mechanism.	• Fail to consider a proper consensus operation like chaincode to validate the design requirements of IoT applications. • Does not apply any practical scenario to validate the users' privacy.
Patwary et al. [17]	Using blockchain technology to present a decentralized framework that can independently authenticate IoT devices.	Secure Decentralized D2D Authentication Model with Location Privacy.	• Fail to apply formal and informal security models in order to identify ambiguities or inconsistencies in the proposed mechanism. • Does not mention any specific consensus protocol to validate the metric values like execution and transaction cost.
Wang et al. [18]	Proposed a cloud-assisted framework to analyze system features such as security and privacy.	Secure and Privacy Preserving Protocol.	• Fail to preserve the privacy of user identity while performing the re-encryption of ciphertext to specify the information of data requester. • Does not apply any specific network scenario using consortium blockchain to guarantee access control and system efficiency.
Karumba et al. [19]	Constructed an adaptive framework to address the issues of peer-to-peer trading systems.	Data Tagging and Anonymization.	• Fail to prove a feature of scalability based on the blockchain adaptive model to evaluate the issue of interoperability while validating the values of cross-network transactions. • Does not apply any key encryption scheme to validate the users' privacy.
Samir et al. [20]	Presented a trustworthy framework to preserve the identities of IoT devices.	Decentralized Identity Management with Trustworthiness.	• Fail to prove the robustness of identity management framework against probabilistic-based security model. • Does not apply any specific consensus mechanism to verify the stages of the trustworthy framework.
Y Zhang et al. [21]	Presented an efficient privacy-preserving authentication based on blockchain to minimize the on-chain storage cost.	Multi-factor Key Derivation Using Physical Unclonable Function.	• Fail to apply a proper security model to analyze the efficiencies of inter and intra-domain authentication. • Does not have any proper consensus mechanism to validate the authenticity and integrity of multi-factor authentication.
J Zhang et al. [22]	Developed a conditional privacy preserving authentication framework to meet the security requirements of VANET such as distributed message authentication and anti-single-point failure.	Privacy Preserving Authentication.	• Fail to consider a proper consensus operation like chaincode to validate the design requirements of vehicle ad-hoc networks while evaluating the solution of decentralized authentication. • Does not apply any practical scenario to validate the users' privacy.

A. Elliptic Curve Cryptography

Assume k is a perfect integer that applies over the projective closure of a non-singular curve to define *'Weierstrass'* equation.

$$y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_6. \quad (1)$$

In Eq.1, a_1, a_2, a_3, a_4, and a_6 are the constant integer fields. When the characteristic k is not of the field 2, the Eq. 1 can be rewritten as,

$$\left(y+\frac{a_1x}{2}+\frac{a_3}{2}\right)^2=x^3+\left(a_2+\frac{a_1^2}{4}\right)+ \left(a_4+\frac{a_1a_3}{2}\right)+\left(a_6+\frac{a_3^2}{4}\right). \quad (2)$$

In case of restricting the projective closure, Eq. 2 can be further simplified as follows.

$$y^2=x^3+Ax+B. \quad (3)$$

The Eq. 3 shows that it can be non-singular, if and only if x^3+Ax+B has its own distinct root over $\overline{k}$ to hold the quantity $\Delta:=-16(4A^3+27B^2)$ (i.e., non-zero) for the constant factors A and B.

Definition 1. *Elliptic Curve Discrete Logarithm Problem (ECDLP).* Elliptic-curve E_C uses an additive group point G to define its prime integer p. For any group point Q of G, we can obtain $Q=kG$ where $k\in Z_p$. It is worth noting that the parameter k is hard to find, even if the values of G and Q are revealed.

TABLE II: Notation Used

Parameter	Description
N_a	Address of the network manager
C_a	Address of the clients including patient and doctor calling the function
$M_a \to F(P)$	Network manager executes the function F with a transaction parameter P
$C_a \to F(P)$	clients including the patient and doctor call the function F with a transaction parameter P
A_W	Array of whitelisted addresses
A_C	Array of created smart contracts
A_R	Array of created transaction requests
A_P	Array of addresses of the clients
P_c	Number of clients approved the transaction or connection requests
E_N	Enterprise Network
ID_{E_N}	An unique identity of E_N
P_A	Partnership Amount
T_A	Trusted Authority
Charlie and Ben	Contract Signer Peers
DT	Medical Data
C_{ID}	Client Identifier
P_{ID}	Proposal Identifier
S_C	Smart Contract
a,b	Random Integers
p	Prime Integer
G	Cyclic additive group
pub_{key}	Public Key
pvt_{key}	Private Key
r,s_j	Pair of integers where $j = 0,1,2,...$
N_{um}	Network authorities
δ	duration
τ	execution index
DC	digital currency

B. Elliptic-Curve Digital Signature Algorithm (EC-DSA)

Elliptic-curve cryptography (ECC) uses elliptic-curve mathematics to design a robust public-key encryption algorithm that utilizes smaller key lengths to offer a better security level than Rivest-Shamir-Adleman (RSA) encryption. Moreover, the designed EC-DSA utilizes a key update algorithm to ensure the property of forward security.

Key parameters: Assume that the execution time of the system is Partitioned into T periods to define a finite field $GF(p)$ with a prime integer p. The system considers an elliptic curve $E_C : Y^2 = x^3 + ax + b \mod p$ over the finite field $GF(p)$ where $(a,b \in GF(p), 4a^3 + 27b^2 \neq 0)$. The elliptic curve with order n defines a base point of G i.e., $Ord(G) = n$ where n is a higher-order prime integer to represent a composite number $N < n$. This applies $h = \#E_C(GF(p))/n$ with a co-factor $(h \ll n)$ to represent a point on the elliptic curve $\#E_C(GF(p))$. Additionally, a secure hash function $H : \{0,1\}^* \to \{0,1\}^{256}$ is applied to define a string with any size of the key length (i.e., to create a client identifier C_{ID}). The initial private and public keys $pvt_{key} = a_0(a_0 \in \{1, n-1\})$ and $pub_{key} = a_0^{-1}.G$ are chosen to perform encryption/decryption process in a network. Lastly, except the private key pvt_{key_0}, the other elliptic curve parameters $p_d = \{p,a,b,G,n,h(.)\}$ including pub_{key} and $H(.)$ are exposed over a wireless channel to establish a secure communication among the computing peers.

Key Update: Assume a j^{th} period of the execution time i.e., $(1 \leq j \leq T)$, the contract signer chooses $pvt_{key_{j-1}}$ of the time period j to compute $pvt_{key_j} = pvt_{key_{j-1}} \mod N$. As a result, the j^{th} period of the private key becomes $pvt_{key^j} = a_0^{2^j}$, where $a_0^{2^j}$ is a pre-computed key of each peer session. It is worth noting that the system always removes the key $pvt_{key_{j-1}}$ in order to maintain the current key pvt_{key_j} in the key generated list.

The computation process including signature and verification is as follows:

Signature Process: The contract signers like Ben and Charlie wish to sign a new message N_{msg} which publishes the system parameters $\{p, a, b, G, n, h()\}$ using ECC to perform encryption/decryption process over j^{th} period i.e., $(1 \leq j \leq T)$. The computation processes are as follows:

Step 1: Charlie finds a random integer i.e., $k \in [1, N-1]$ to generate a public key $pub_{key} = pvt_{key}.G$ where pub_{key} and pvt_{key} are Ben's public key and the private key.

Step 2: Accordingly, Charlie chooses a random integer $k^{2^{-j}}$ i.e., $[1, N-1]$ to compute the coordinates of the elliptic-curve: $(x_1, y_1) = k^{2^{-j}}.G$, $z = h(N_{msg})$, $r = x_1 \mod n$, and $s = (z + pvt_{key}.r).K^{-1} \mod n$. In case $r = 0$, re-execute the process of *Step 1*.

Step 3: Lastly, Charlie finds a hash value $e_j = H(j, N_{msg}, r)$ and $s_j = pvt_{key_j}.k - e_j \mod N$. In case $s_j > 0$, Charlie transmits the signature results $sg_j = r, s_j$ to Ben. Otherwise, Charlie returns to *Step 1*.

Verification Process: Ben wishes to verify whether the signature (r, s_j) issued by Charlie contains a legitimate message N_{msg} over a time period $1 \leq j \leq T$. The execution steps are as follows:

Step 1: Initially, Ben uses $r, s_j \in [1, N-1]$ to verify the following computation: $z' = h(N_{msg})$, $u_1 = z'.s_j^{-1} \mod n$, $u_2 = r.s_j{-1} \mod n$, and $(x_1', y_1') = u_1.G + u_2.pub_{key}$. In case of an unsuccessful, Ben rejects the signature issued by Charlie.

Step 2: After successful verification, Ben finds the hash value $e_j' = H(j, N_{msg}, r)$, $w_j = e_j' + s_j \mod N$, $X = w_j^{2^{-j}}.pub_{key} = (x_2, y_2)$ to check whether $X > 0$ or not. In case $X = 0$, Ben rejects the signature of Charlie.

Step 3: Finally, Ben finds $v = x_2 \mod N$ to check whether $v = r$ is valid or not to show its equality with the issued signature. In case of an unsuccessful verification, Ben rejects the issued signature of Charlie i.e., $S_j = (r, s_j)$.

C. Consensus Mechanisms

A few constructive consensus mechanisms are categorized as proof-of-work (PoW), proof-of-stake (PoS), delegated proof of stake (D-PoS), and proof-of-trusted authority (POTA) to verify and maintain the security of the blockchain networks. Moreover, a constructive mechanism so-called PoA is adopted to analyze the limitations of the network participants, including quality of experience (QoE). Network participants comply with agreed-upon rules [26] to ensure the transaction process is legitimate. To regulate the process, blockchain technology authorizes network participants. Therefore, public or private networks demand user authorization to authenticate the network activities that make transactions more reliable and secure to allocate the computing resources evenly to blockchain Mempool.

Proof of Trusted Authority (POTA): This technique selects its validator based on its reputation value to generate a new block. Each block validates its own transaction process via a trustworthy feature to protect the blockchain. Moreover, the validator uses certain conditional procedures to ensure key properties such as reliability and integrity. The conditional procedure authorizes transaction

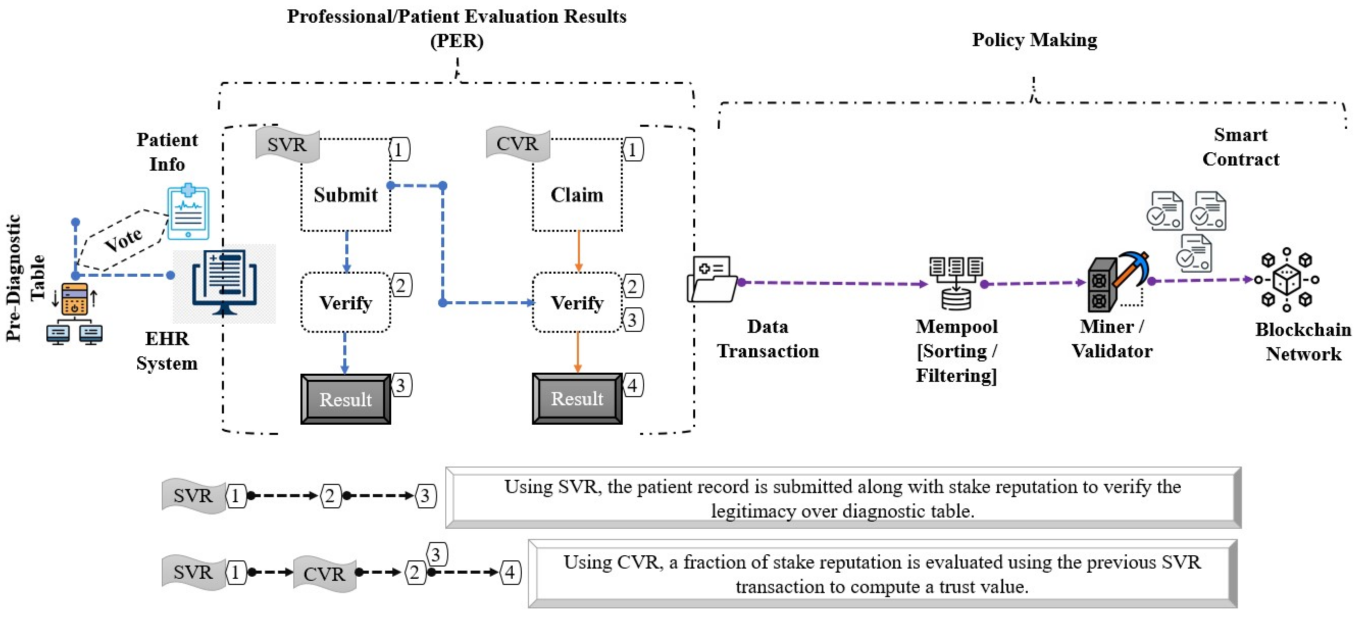

Professional/Patient Evaluation Results (PER)
Policy Making
Patient Info
Vote
Pre-Diagnostic Table
EHR System
SVR
Submit
Verify
Result
CVR
Claim
Verify
Result
1
2
3
4
Smart Contract
Data Transaction
Mempool [Sorting / Filtering]
Miner / Validator
Blockchain Network
Using SVR, the patient record is submitted along with stake reputation to verify the legitimacy over diagnostic table.
Using CVR, a fraction of stake reputation is evaluated using the previous SVR transaction to compute a trust value.

Fig. 1: A Model of POTA Using Vote-Per-Policy

blocks to mine large numbers of coins, which consumes more energy than other network participants [37]. It applies a peer-coin strategy, taking into consideration the randomized process that creates the blocks to maintain the amount of cryptocurrency. Moreover, one specific benefit is to minimize the cost of the mining process so it does not require any network participant to examine transaction activities. On the other hand, this technique allows the stakeholder to witness a vote on the transaction process. It minimizes energy storage to improve the computation speed of the transaction. Hence, the overall block generation makes the transaction process faster than PoW. Typically, a strategy with a vote-per-share policy provides vote casting to the stakeholder, whereby more coins may be possessed.

As a result, the promising solution uses a contextless transaction to initialize the medical diagnosis and treatment process in an unsecured environment. This environment deals with unconfirmed transactions residing in a blockchain Mempool where each participant sets its own in-memory data structure over a dedicated Ethereum network to perform transaction ordering, fee prioritization, and block construction. In other words, valid contextless transactions are initiated via decentralized applications (dApps) which use proper signature mechanisms to gain Ethereum network access in order to add a new block across the network. Considering a crypto-economic model, the interaction among the users and smart contracts verifies a number of coins to mean a stake in the reputation given to honest peers to determine the confiscated coins. As a consequence, in the EHR system, two different entities such as medical experts and patients are chosen as the trusted ones. However, the Internet-enabled devices of the entities i.e., IoT are prone to various security risks (i.e., insecure passwords and lack of encryption) due to severe data breaches associated with poor access control. Therefore, the POTA logically applies the transaction models including *submit-verify-result (SVR)* and *claim-verify-result (CVR)* to facilitate the process of vote-per-share policy based on the representation of the workflow (as shown in Fig.1).

Using this representation, the transactions such as *SVR* and *CVR* work with Mempool to initialize a contextless voting-per-share system to verify any pre-populated diagnostic table. The diagnostic table collects the patient's information shared by medical experts along with the staked reputation to compute the trust value. In case of successful computation, the state of the related records is accordingly executed without smart contract evaluation to create a new block which is later embedded in the blockchain network to specify enough gas limit as required by each participant. Moreover, in consortium chains, POTA includes various enterprises where the network features of private and public chain use permissioned blockchain to share a ledger among the IoT networks. As the IoT node has limited computation capacity, the POTA designs a lightweight transaction model with computation insensitivity to execute tasks or applications with *Zero Trust*. The main idea of this model is to set a context to the supplied transaction which differentiates the priority via a Mempool of the participants i.e., a temporary queue in order to operate the system with minimum overhead.

Importantly, this model allows the stakeholder to witness a vote on the transaction processes via PER i.e., professional/patient evaluation results to minimize energy storage whereby the computation speed of the transaction is improved. Hence, the overall block generation makes the transaction process faster than other consensus mechanisms like PoW and PoS. Typically, a strategy with a vote-per-share policy provides vote casting to the stakeholder, whereby more coins may be possessed. Above all, the transaction models (SVR and CVR) maintain better consistency with the Ethereum network to examine the behavior of neighbor nodes linking to the same IoT networks in case of evaluating their credibilities including authentication, authorization, and validation.

D. Hyper-Ledger Fabric

This open-source blockchain framework by The Linux Foundation promotes cross-platform development. It does not reflect the features of digital currency in order to verify system reliability and performance efficiency. However, it uses a practical Byzantine Fault Tolerance (pBFT) mechanism to improve the abilities of the consensus algorithm [38]. The decentralization innovation applies pBFT to promote consensus mechanism and has a wide range of modular design and architecture to achieve significant features such as adaptability and flexibility. The design architecture uses hyper-ledger fabric to process four functional management systems namely identity, ledger, transaction, and smart contract which explore consensus, security, and cryptographic services to host blockchain network and execute the data blocks via distributed ledger to peer/client nodes. The fabric network consists of six blockchain components to examine the system logs and they are as follows.

Cloud/Network Service Provider: This is a type of managed service provider (MSP) that validates and authenticates the rules defined by the networks.

Smart Contract: In Hyperledger Fabric, a smart contract generates a chain of codes (chaincode) to define the system assets and their related data transactions. Each chaincode interacts with a distributed ledger to identify the signed transactions rendered by smart contracts.

Ordering Service: With this service, network packages are ordered to deliver channel peers that deal with transaction histories over a public network.

Client Identities: Each client manages the network peers, clients, and the system manager to generate a digital identity with the given format: X.509. It manages the system logs to validate whether a transaction is legal or not.

Network Channels: Multiple channels organize system access to maintain legal transactions between the multiple blockchains.

Peer/Client Nodes: These constitute the elements of the network to configure smart contracts with the distributed ledger.

Fig. 2 shows the specific transaction flow strategies in hyper-ledger fabric and are illustrated as follows:

1) *Flow 1 - Initiating the transaction:* The patient/doctor so-called client is the initiator of the data transaction. They invoke a chaincode function to create a proposal request. This function applies on a network channel to accept the proposal request signed and submitted by the client.
2) *Flow 2 - Executing the transaction:* After receiving the data transaction, the service providers verify the legitimacy of the signed transaction via distributed ledger to authenticate the proposal request. To prove its legitimacy, the service provider validates a few common privileges including authenticity, authorization, and replay protection.
3) *Flow 3 - Collecting the ordered proposal request:* The clients receive the proposal requests confirmed by the service provider to compare and verify the requirements according to the policy of the vote-per-share function.
4) *Flow 4 - Validating and Committing the transaction:* The service provider issues the ordering requests in the chain of blocks to all the computing peers/clients over a dedicated channel. According to the policy of the vote-per-share function, the computing peers/clients verify the transaction before it is added to the distributed ledger.

1) Generated Chaincode: In Hyper-ledger Fabric, the chaincode encapsulates data schema to create and update the business logic while any contract peer triggers the ledger to process the payment to the EHR system. The ledger prefers different programming languages like Node.js, Go, and Java not only to enforce the business logic but also to facilitate proper authentication and execution among the computing peers to update the status quo of the ledger. It is also evident that the chaincode performs a few specific tasks including reading, executing, and modifying to generate the peers' copies while any conditional logic triggers the ledger to submit the transaction to the blockchain network.

2) Associated Inter-Planetary File System (IPFS): The associated IPFS acts as a peer-to-peer distributed file system that uses distributed storage service to process storage content corresponding to the unique hash code 31. It is worth noting that the unique hash code so-called content identifier (CID) links the hash of the file to derive a string address. This address makes the computing peer to participating independently without any trusted third parties or causing a single point of failure like traditional *HTTP* transfer. As a result, data access prefers its own neighbor node to accelerate the process of data transfer with minimum storage overhead. Moreover, in the IPFS, the peer-to-peer transmission keeps network bandwidth and distributed file contents in reserve via a distributed hash table to prevent potential attacks like distributed denial of service (DDoS) whereby the features such as high throughput and data anti-tempering can be achieved.

E. System Model

Fig.3 illustrates an overview of the high-level cloud-based consortium blockchain architecture. It comprises four significant entities: the patient, the cloud server, the medical center, and the system administrator. Details are as follows.

Patient: Physiological data of the medical patient are sensed and transmitted to the registered medical center over a secure channel. This channel uses the generated keys i.e., pub_{key} and pvt_{key} of the computing peers to process the application data securely to the registered medical center. While preserving its confidentiality and integrity, the computing peers can periodically collect and record the health data in an EHR provided by the registered medical center.

Medical Center: In a private blockchain, the registered medical center uses a *Whitelist* to control the access of the participants by system administrators to generate and record the EHRs in the cloud server along with medical center information.

System Administrator: This is a trusted third party to maintain participants' registrations and manages the consortium blockchain.

Cloud Server: This is a trusted entity entitled to maintain power and resource efficiencies and to maintain and store health data to achieve data-sharing and resource efficiency.

Hyper-Ledger Network Layer: This layer comprises computing peers/clients, ordering service nodes, network channels, and certificate authority which uses certificate authority (CA) to issue digital certificates, public and private keys. CA authenticates administrators and computing peers to gain a blockchain network that uses a dedicated channel to share the data via a private blockchain. The network associates with the ledger and peer via its own channel to ensure data sharing between diverse enterprises while protecting the privacy of data. Each channel tries to access the ordering service node which collects and broadcasts the pack of transactions to the computing peers.

Client and Storage Layer: Each enterprise has an administrator to offer seamless interaction with the fabric network. The network

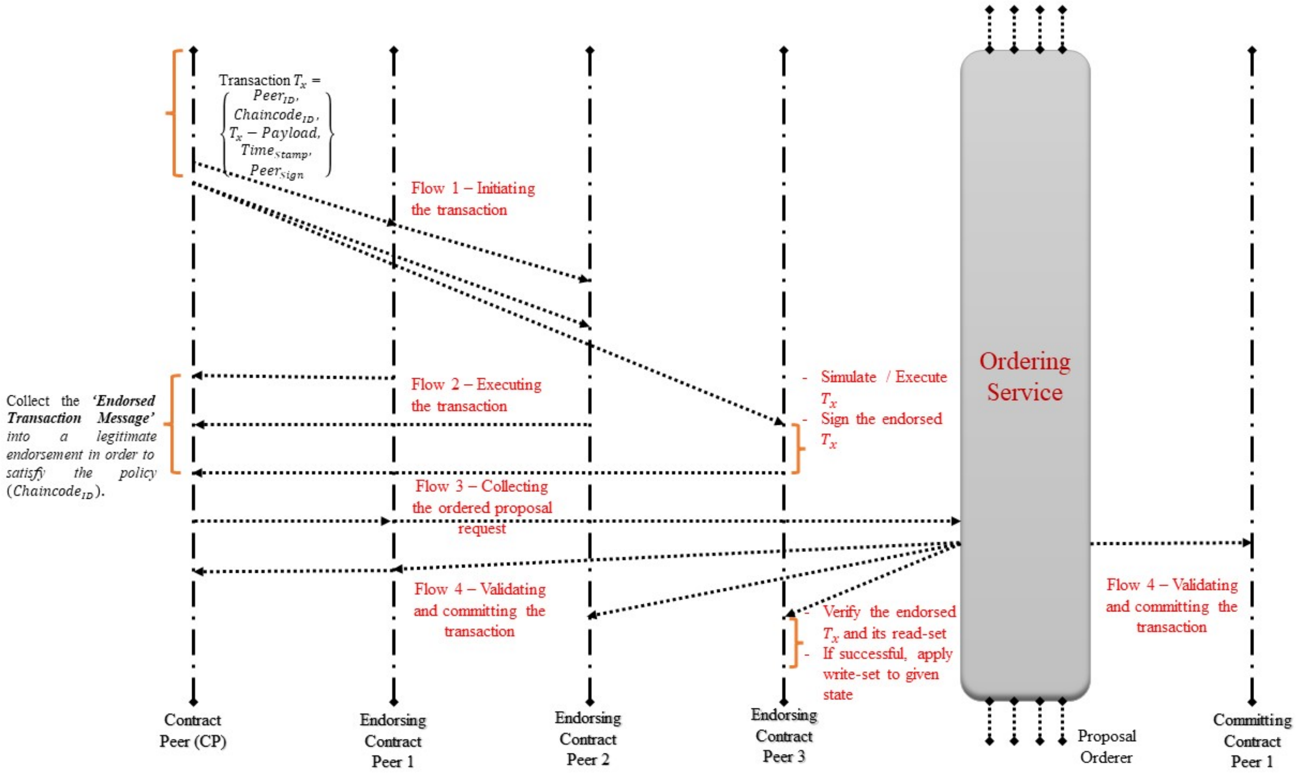

Transaction T_x = {$Peer_{ID}$, $Chaincode_{ID}$, $T_x - Payload$, $Time_{Stamp}$, $Peer_{Sign}$}
Flow 1 – Initiating the transaction
Collect the 'Endorsed Transaction Message' into a legitimate endorsement in order to satisfy the policy ($Chaincode_{ID}$).
Flow 2 – Executing the transaction
- Simulate / Execute T_x
- Sign the endorsed T_x
Flow 3 – Collecting the ordered proposal request
Ordering Service
Flow 4 – Validating and committing the transaction
- Verify the endorsed T_x and its read-set
- If successful, apply write-set to given state
Flow 4 – Validating and committing the transaction
Contract Peer (CP)
Endorsing Contract Peer 1
Endorsing Contract Peer 2
Endorsing Contract Peer 3
Proposal Orderer
Committing Contract Peer 1

Fig. 2: Transaction Flow Strategies in Hyper-Ledger Fabric

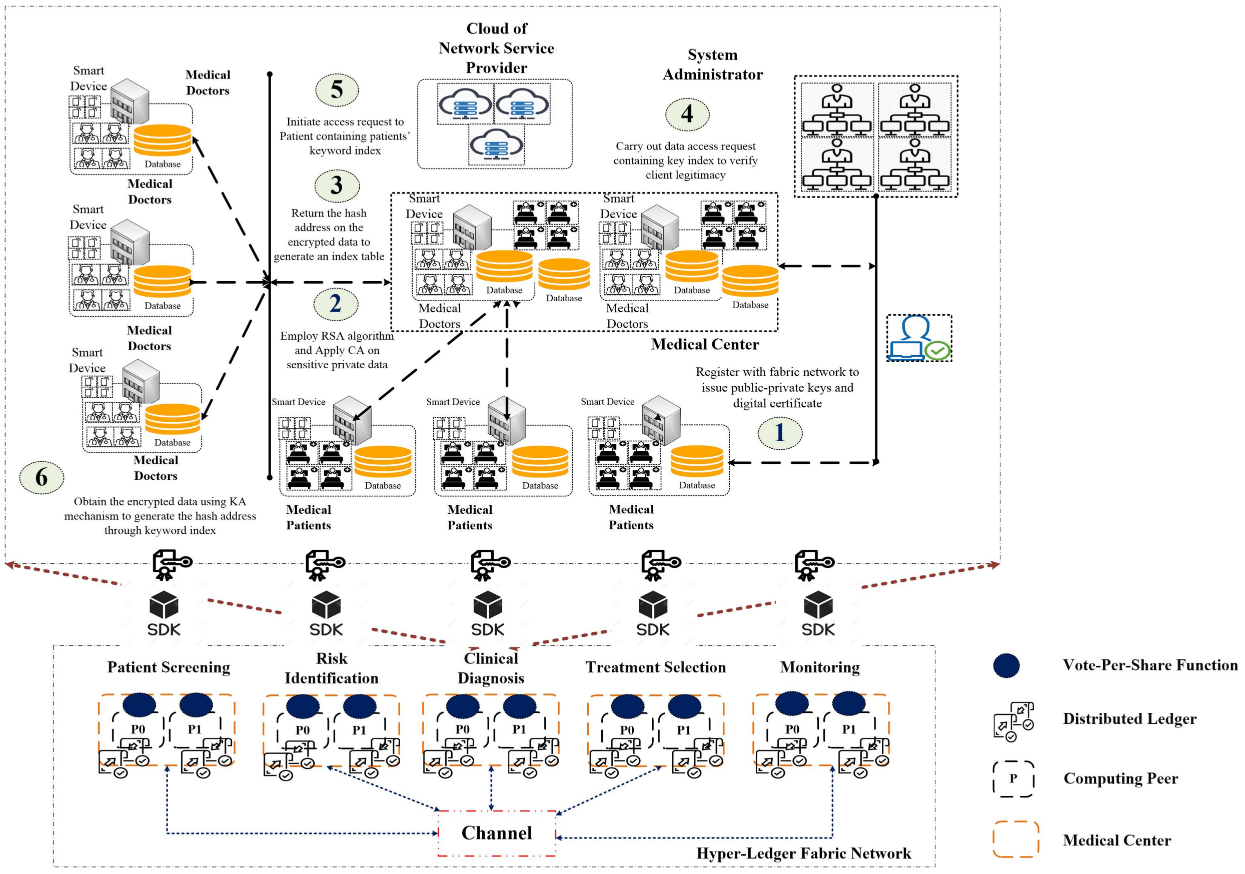

Cloud of Network Service Provider
System Administrator
Smart Device
Medical Doctors
Database
5
Initiate access request to Patient containing patients' keyword index
3
Return the hash address on the encrypted data to generate an index table
2
Employ RSA algorithm and Apply CA on sensitive private data
4
Carry out data access request containing key index to verify client legitimacy
Medical Center
Register with fabric network to issue public-private keys and digital certificate
1
6
Obtain the encrypted data using KA mechanism to generate the hash address through keyword index
Medical Patients
SDK
Patient Screening
Risk Identification
Clinical Diagnosis
Treatment Selection
Monitoring
P0
P1
Channel
Hyper-Ledger Fabric Network
Vote-Per-Share Function
Distributed Ledger
Computing Peer
Medical Center

Fig. 3: The High-Level Architecture of the Consortium Blockchain and the Smart Contract.

prefers a software development kit (SDK) to connect the administrator with the blockchain network through computing peers. Using the vote-per-share policy, the administrator registers the computing peer through CA to participate in the process of data transaction.

The enterprise connects with the channel of the computing peer via a distributed file system to store and share the data transaction. The system administrator generates a hash address using the KA mechanism to secure data transactions and constructs an index table to upload it to the blockchain network. For data sharing, the fabric network configures multiple channels and enterprises. The administrator of the enterprise creates its own CA in the blockchain network which applies a public-private key and a digital certificate using the X.509 standard to endorse the decree of data transaction. The cloud-assisted decentralized applications including patient/medical doctors and medical centers access the same channel to offer a reliable data transaction to the clients and the workflows are as follows.

1) Patient/Medical Doctor registers with the fabric network through the computing peers to issue public-private keys and digital certificates in order to complete the registration phase.
2) Medical Center employs *RSA* encryption algorithm to encrypt the system parameters symmetrically and applies CA on the sensitive private data before being saved onto the fabric network.
3) The fabric network returns the hash address on the encrypted data to generate an index table that executes the vote-per-share policy via the blockchain network to complete the phase of data storage.
4) Patient carries out data access request containing keyword index to verify the client legitimacy via a directory of EHR and return the hash address in fabric network.
5) Medical Doctor initiates the access request to Patient containing patients' keyword index which uses public-private keys to generate an encrypted message.
6) Patient/Medical Doctor obtains the encrypted data using the KA mechanism to generate the hash address through the keyword index in order to obtain the original source data.

IV. The Proposed CA-DPPF Using Blockchain

As a solution to the problems in the existing system, a decentralized approach is proposed with a consortium blockchain, cloud computing, and smart contracts. The participating entities are the network platform, the peer holder, a manager (i.e., evaluator, authority, and verifier), and a set of clients. The system consists of a decentralized network platform (DNet) that provides the interface to the whole system. The DNet allows the participating entities (the network platform, cloud, manager, and clients) to interact with the Hyperledger Fabric platform. Each action performed in the DNet by the entities is a corresponding function of the smart contract executing the transaction stored in the consortium blockchain through a proper KA mechanism.

The system has two-layer consortium blockchains with three computing peers that apply a consensus mechanism i.e., POTA to validate the process of transaction as shown in 4. It uses a verifiable repository to seal the transaction [39]. The smart contract has seven functions (i.e., create, upload, search, check, assess, verify, and submit) categorizing the features of decentralized systems. The main objective of the system is to separate manager (i.e., evaluator, authority, and verifier) responsibilities from the central authority establishing the entities creating contracts. The clients must be associated with a partnership to validate any proposal process under a contract. Moreover, the clients contribute P_A to verify the validity of the client interaction. The manager generates a payment request to the medical center, which regulates the request created by the manager. It must adhere to decentralized properties that create a request to claim the transaction amount required for a successful P_A transfer.

The clients execute the recommended function to approve or reject the proposal request based on the integrity of the requirement. Moreover, they permit the manager to transfer the funds to a trusted authority. Lastly, the authority validates the transfer function via smart contract. In order to improve the privacy and speed of transactions, the contract can be deployed to a consortium chain, which has three nodes, the first node being the manager and the other two being clients of the organization. Initially, DNet verifies the certification process issued by a trusted authority to find whether the client identifier C_{ID} is recorded in the blockchain network. In case of availability, DNet searches and collects the proposal identifier P_{ID} to execute the data transaction via a function of smart contract shown in Algorithm 1.

Algorithm 1 Verify the Issuance Certification

```
Require: $(P_{ID}, C_{ID})$
Ensure: Executing the state of the transaction.
1: $P \leftarrow S_C.Get-Certificate_{Info}(C_{ID})$
2: if $P == Null$ then
3:     $P \leftarrow S_C.Get-Proposal_{Info}(P_{ID})$
4:     if $P == Null$ then
5:         Return 'No Certificate Issued'
6:     else
7:         $S_C.Issue-Proposal_{Cert}(P)$
8: else
9:     Return 'Certificate Issued'
```

A. Algorithms

In this section, algorithmic functions such as creating enterprise, treatment, payment, management, and transfer are constructed to improve flow efficiencies in data transmission. Table II shows the important notations used in the given algorithms.

Algorithm 2 shows the function of the manager to create an enterprise and create a smart contract. This function takes an input P_A, which specifies the minimum value of P_A being paid by the client to establish a smart contract in order to participate in the decision-making process.

Algorithm 2 Create a Enterprise Network E_N

```
Require: Trust Amount $A_W$
Ensure: The address of the created contract is added to the array of smart contracts $A_C$
1: Read Enterprise amount, $A_W$
2: if $C_a$ in $A_W$ then
3:     if $C_a == M_a$ then
4:         $C \rightarrow$ Create Healthcare Network()
5:         Start Bid
6:         require ($M_{sg}$.sender== Service Agent)
7:         $bid_{started} \leftarrow True$
8:         End Bid
9:         $A_C \leftarrow$ Address of $S_C()$
10:        Start Reveal
11:        require ($M_{sg}$.Sender == Service Agent)
12:        $bid_{started} \leftarrow False$
13:        $reveal_{started} \leftarrow True$
14:        End Reveal
15:    else
16:        Transaction cannot be executed.
17: else
18:    Transaction cannot be executed.
```

Before creating the smart contract, there is a modifier function to verify whether the caller address is available in the whitelist addresses A_W or not in order to execute a transaction in the blockchain. In case of successful verification, the service agent finds a suitable auction for the patient in order to call the bid function in the public blockchain *[Algorithm 2: Lines 4.1 to 4.4]*. In the reveal phase, the bid function commits to the private blockchain *[Algorithm 2: Lines 5 to 8]* which reveals whether the patient has a bid available in the public blockchain. Otherwise, the private blockchain removes the patient from the whitelist. It is worth noting that this procedure makes the patient to mutilating the interaction with the blockchain to prevent security threats and data breaches.

Algorithm 3 Initialize the Treatment Process

```
Require: Trust Value, $A_W$
Ensure: The addresses of clients (Patients) are added to the client's array (Doctors).
 1: Read Trust Value, $A_W$
 2: if $C_a$ in $A_W$ then
 3:     if (Trust Value == Trust Amount) then
 4:         $A_P \rightarrow C_a$
 5:         Start Make Commitment (Truster, $A_W$)
 6:         require ($M_{sg}$.Sender == Service Agent)
 7:         require ($reveal_{started} == True$)
 8:         bids(Truster).$push \rightarrow A_W$
 9:         End
10:     else
11:         Transaction cannot be executed.
12: else
13:     Transaction cannot be executed.
```

Algorithm 3 shows the processes of the treatment function. The objective of this function is to contribute P_A to the contract. If the monetary contribution is greater than the trust amount specified by the manager during smart contract creation, then the client's address is automatically added to the client array to participate in the decision-making process [40]. The patient bids on the treatment process to begin the proposal auction, which requires the commitment scheme to involve the bidder. It can process a random number r and the trust value, A_W, from each patient *[Algorithm 3: Lines 4 to 9]* that executes a payment process under a smart contract to confirm the transaction.

Algorithm 4 Approve the Payment Process

```
Require: Trust Value, $A_W$
Ensure: Requests are added to the array of clients' requests.
 1: Read Trust Value, $A_W$
 2: if $C_a$ in $A_W$ then
 3:     $A_R \rightarrow$ Request ()
 4:     Reveal Commitment ($A_W$,$random_{value}()$)
 5:     require ($reveal_{started} == True$)
 6:     $bid_{length} \leftarrow bids(M_{sg}.sender).length$
 7:     require ($bid_{length} == A_W().length$)
 8:     require ($bid_{length} == random_{value}().length$)
 9:     for $k \rightarrow 1$ To $A_W().length$ do
10:         if bid($M_{sg}.sender[k]$) then
11:             Bid Commitment $A_W(k)$
12:             $true_{Amount} \rightarrow A_W(k)$
13:             $false_{Amount} \rightarrow false_{Amount} + A_W(k)$
14:             Emit Penalize Event
15:             Check $IF_{Winner}(M_{sg}.sender, A_W)$
16: else
17:     Transaction cannot be executed.
```

Algorithm 4 shows the process for payment requests. This function creates the payment request approved by the client. The request details, including an explanation and the expensed amount, are required to complete the transaction request. The address of the third party receives the transferred amount after successful finalization by the requester. As a result, the requester's details can be added to the requested array as and when the payment request is created. In the reveal commitment, an event such as $reveal_started$ maintains a bid at the patient's request in order to execute the smart contract function. It uses a one-way permutation bit to modify the commitment protocol that encapsulates a hash value to record the public blockchain *[Algorithm 4: Lines 3 to 15]*.

Algorithm 5 Management Function

```
Require: Trust Value, $A_W$
Ensure: Requests are added to the array of clients' requests.
 1: Read Index, Trust Value, $A_W$
 2: if $C_a$ in $A_W$ then
 3:     if $C_a$ in $A_P$ then
 4:         Approve Request
 5:         $P_C \rightarrow P_C + 1$
 6:         Check $IF_{Winner}(Truster, A_W)$
 7:         if ($A_W \leq highest_{Trusted}$) then
 8:             if ($A_W \geq SecondHighest_{Trusted}$) then
 9:                 $SecondHighest_{Trust} \leftarrow A_W$
10:                 $SecondHighest_{Trust} \leftarrow Trusted$
11:                 return True
12:             else
13:                 $SecondHighest_{Trust} \leftarrow highest_{Trust}$
14:                 $highest_{Trust} \leftarrow A_W$
15:                 $SecondHighest_{Trusted} \leftarrow highest_{Trusted}$
16:                 $highest_{Trust} \leftarrow Trusted$
17:                 Emit Bid Event
18:         else
19:             Transaction cannot be executed.
20:     else
21:         Transaction cannot be executed.
```

Algorithm 5 shows the management function processed by the clients. The main objective of this function is to approve the requests associated with the manager. In order to approve a request, the client's address is required to call the function. It should be available in the client's address array. Upon approval of the client's request, a counter keeps track of the client's availability in order to increment the approval payment by 1. In the reveal phase, the trusted patient bids on two events (the first winner and the second winner) to create a distributed ledger *[Algorithm 6: Lines 3 to 15]*. The management process executes the intermediary functions to link the consortium blockchain that controls the data capture to execute the treatment phase again. *Algorithm 6* shows the manager transfer function.

Algorithm 6 Transfer Function

```
Require: Trust Index, $T_I$
Ensure: Partnership amount $P_A$ is transferred to a third-party account.
 1: Read Index, Trust Value, $A_W$
 2: if $C_a$ in $A_W$ then
 3:     set $highest_{Trust}(A_{W1}, A_{W2})$
 4:     require ($Msg$.Sender $\neq$ Service Agent)
 5:     Trusts(Trusted).push $\leftarrow A_W$
 6:     End
 7:     if $C_a$ in $A_P$ then
 8:         set $highest_{Trusted}(A_{W1}, A_{W2})$
 9:         commitments ($transfer_{failed}$)
10:         Trusted Commitment ($transfer_{failed}$)
11:         if $P_C(Index) == 2$ then
12:             Finalize the request
13:             Make the request as complete
14:         else
15:             wait till $P_C = 2$
16:     else
17:         Transaction cannot be executed.
18: else
19:     Transaction cannot be executed.
```

B. Creating a Smart Contract

The creation of a smart contract uses the programming language known as Solidity. In order to satisfy the requirements of the proposed system, it has four important functions as follows:

1) The manager creates the partnership function, which gives P_A to be paid by the clients to an enterprise network. Moreover, P_A represents the number of assets being traded on the supply of partnership to access the blockchain in order to return its value on the ledger.
2) The contribution amount is paid to the third party upon successful completion of a payment request. Moreover, the third party may use P_A to track and capture the status of the request via a dedicated IoT device.
3) The manager creates the payment request to handle the details of the transaction request. In consequence, the security of the ledger and its accessibility type can be maintained remotely to authenticate the transaction i.e., using the KA mechanism.
4) The clients include the purpose of a client request or demanded P_A to complete any transaction request that signifies the use of a third-party wallet, plus payment completion status. On the other hand, the clients call the pay function with a value i.e., P_A, which should be greater than the minimum contribution. Upon verification, the client pays the amount greater than P_A allowing participation in the decision-making process [41].

C. Creating a New Peer using KA mechanism

The registration process involves an enterprise or private network to include the blockchain network that sets up full network connectivity to the entities, namely, a patient and an expert. Upon connection of the blockchain network, the functional nodes generate virtual wallets and device identities, including public and private keys. After the successful generation of the identities, the entities contact delegates through the blockchain network to transmit the registration request. Subsequently, the delegates validate the registration request to transfer a legitimate token available in the blockchain address. The role of delegate authorization is to witness a vote on the transaction process which allows the stakeholder to use an access control policy in order to verify the clients' registration request whereby the computation speed of the transaction is improved to minimize energy storage.

As a result, a connective address is added to the blockchain network to broadcast the registration details of peer holders. The assigned delegate allows the network peers to obtain the wallet information to process the business transaction. Additionally, a new client renders the network instruction to set up the client as an elected delegate to verify the transaction cycles: token size, blockchain address, and signed public and private keys. After successful validation, the client may execute the transaction to include a block. Once the process is complete, a new network may set up a process to broadcast the selected network as a delegate to enhance the data encryption across the public/private network[42]. Thus, even if an intruder registers as a legitimate client in the network region, it cannot change the transmission data in particular.

Peer Registration The client registers the physical network devices to visit the enterprise agencies that issue valid IDs for physical entities. The delegate discovers an authentic blockchain address to associate with valid private and public keys. The address is useful when inferring the identity of the owner. Accordingly, the entities obtain a blockchain wallet to broadcast details of the blockchain address that maintains the transaction details of the individual client. Moreover, confidential details such as ID_{E_N}, pub_{key}, and pvt_{key} are safely stored in the given database. It is worth noting that this associated database creates a dedicated wallet interface to protect the digital assets of the clients which rely on the features of asymmetric encryption and tamper resistance using the KA mechanism and blockchain to prevent the risks of key extraction. However, to substantiate the instance-specific behavior of the system i.e., wallet interface, the KA mechanism applies a security assumption based on a probabilistic polynomial-time attacker which controls the communication channel of the clients and E_N to extract their confidential parameters. Lastly, transaction records are maintained to track or view the status of the transactions through the wallet interface. When a client processes any transaction request to a network delegate, it will try to complete the authentication process, whereby the blockchain address can be updated to validate the transaction in the fabric network.

The delegate transfers transaction records including the user identity, the record value, and the registration code to access the user information. The enterprise may use a web application to monitor and protect the blockchain details stored in the common database. Thus, enterprise entities demand a safety backup to prevent acts by any adversary. To achieve better transparency, the enterprise department ensures that entities may create a new blockchain when any of them has lost the private key.

Phase I: Initialization: This phase considers the enterprise network E_N to join the consortium blockchain and later utilizes trusted authority to register the peer holder via DNet. The execution steps are as follows:

1) E_N utilizes the hash function to register the peer information which obtains $h(M_{Submit})$ to send it to T_A.
2) T_A uses an encryption algorithm using EC-DSA to generate pvt_{key} based on the requirement of E_N to compute $pub_{key} = pvt_{key} \times G$. If the identities are genuine, then T_A sends the transmission request (pvt_{key}, pub_{key}) including $Certify_{E_N}$ to E_N client where $Certify_{E_N}$ defines an unique identity ID_{E_N}.
3) Lastly, E_N stocks up the parameters (pvt_{key}, pub_{key}) and $Certify_{E_N}$.

Phase II: Data Storage This phase constructs the hash address of the original data using the KA mechanism (i.e., RSA encryption) through peer holders which construct the hash address of the medical data DT via IPFS and subsequently generate a keyword index table to upload it into the Hyper-Ledger Fabric (HLF) network (as shown in Algorithm 7). The workflow is as follows:

1) 'Peer Holder' defines the name of the enterprise network to perform a few significant computations: $Signature$ is an enterprise network to guarantee data integrity, and ID_{E_N} is a unique identifier of E_N including $Certify_{E_N}$.
2) 'Hash-Address' is the address of any data content (i.e., from IPFS network) to describe the summary of fabric network 'Summary Data' which has a data requester to explore the search content 'keyword' including 'size' and 'type' of the searching data.
3) 'Timestamp' defines the system timing added by peer holders via the blockchain network to discover a unique timestamp value to search the data content.

Step 1: Ben chooses a random integer k_{b1} and the medical data DT to generate a computing message $M_1 = h(ID_B \,\|\, T_1 \,\|\, DT)$. For M_1, the function uses $Signature(M_1, d_b, k_{b1}, A_W)$ which generates a signature $\{r_{b1}, s_{b1}\}$ using encryption algorithm to encrypt the com-

Algorithm 7 Flow Structure of Data Storage.

```
Require: Given Medical Data $DT$
  Generate a message request $M_1$;
  Ben selects a random integer $k_{b1}$;
  Compute $M_1 = h(ID_B \| T_1 \| DT)$;
  Perform signature process on $M_1$: Invoke a function $Signature(M_1,d_b,k_{b1},A_W)$ to return a signature $(r_{b1},s_{b1})$;
  Assign $C_{b1} \rightarrow E_{pvt_{key_{b1}}}$;
  Transfer $C_{b1}$ to IPFS;
Require: Apply hash address;
  Verify whether $T_{now} - T_1 \leq \tau$ or not;
  if $T_{now} - T_1 \leq \tau$ then
      Store the parameter $C_{b1}$ and generate its $Hash_{DT}$;
      Transfer $Hash_{DT}$ to Ben;
Require: Apply $KI_T$;
  Obtain a $KI_T$;
  Ben selects a random integer $k_{b2}$;
  Generate a message $M_2$ and compute $M_2 = h(ID_B \| T_2 \| KI_T)$;
  Perform signature process on $M_2$: Invoke a function $Signature(M_2,d_b,k_{b2},A_W)$ to return a signature $(r_{b2},s_{b2})$; ▷ Initialize the signature function: function Signature (Msg string, dB string, k string) (r string, s string)
  function Signature (Msg string, dB string, k string) (r string, s string);
  $(x,y) = k \times G$;
  $z \leftarrow h(Msg)$;
  $r \leftarrow x \mod N, s \leftarrow k^{-1}(z + r \times dB) \mod N$;
  $return(r,s)$
  Transfer $M_2$ and $(r_{b2},s_{b2})$ to $S_{DB}$;
Require: Add To Distributed Ledger;
  Verify whether $T_{now} - T_2 \leq \tau$ or not;
  Invoke a function $V_{erify}(z_{b2'},r_{b2},s_{b2})$ and return the verified result; ▷ Initialize the verification function: function Verify (z string, r string, s string, qBC string) (output string)
  function Verify (z string, r string, s string, qBC string) (output string);
  $u_1 = z \times s^{-1} \mod N$;
  $u_2 = r \times S^{-1} \mod N$;
  $(x', y') = u_1 * G + u_2 * qBC$;
  if $x' == r \mod N$ then
      Return 'Valid'
  else
      Return 'Invalid'
  if $T_{now} - T_2 \leq \tau$ then
      Call a chaincode in the name of 'File-B' and add the values of the generated parameters $(M_2, Submit_B, A_W)$;
```

puting message. Subsequently, M_1 applies $C_{b1} = E_{pvt_{keyB}}(M_1)$ to store the system parameters $\{r_{b1},s_{b1}\}$ in a file system via IPFS.

Step 2: The file system verifies the timestamp M_1 to resist replay attack and stores the message M_1 in the fabric network to write the hash address to Ben.

Step 3: Ben uses a keyword-index table KI_T to analyze the data keywords and chooses a random integer k_{b2} to compute a message $M_2 = h(ID_B \| T_2 \| KI_T)$. For M_2, the function uses (M_2,d_b,k_{b2},A_W) which generates the signature $\{r_{b2},s_{b2}\}$ using encryption algorithm and sends the parameters $\{M_2,r_{b1},s_{b1}\}$ to system database of Hyper-Ledger Fabric network S_{DB}.

Step 4: S_{DB} verifies the timestamp $T_{now} - T_2 \leq \tau$ to check the function using $V_{erify}(z'_{b2},r_{b2},s_{b2},A_W)$ in order to confirm the legitimacy of Ben's signature. If $(z'_{b2} = r_{b2} \mod n)$ is legal, then the vote-per-share function is executed to add $(M_2,Submit_B,A_W)$ to the distributed ledger $Submit_B = h(r_{b2},s_{b2})$.

Phase III: Data Query E_N generates a query data q_d and submit it to S_{DB} via Charlie. If the generated q_d is legitimate, then S_{DB} returns the respective keyword index to Charlie (as shown in Algorithm 8. In order to execute the workflow, this phase is categorized into two steps.

Algorithm 8 Flow Structure of Data Query.

```
Require: Submit a query data $M_{Charlie \rightarrow S_{DB}}$;
  Generate a query request;
  Charlie selects a random integer $k_{c1}$;
  Compute a query request $M_{Charlie \rightarrow S_{DB}} = h(ID_C \| T_{Charlie \rightarrow S_{DB}} \| key_{words})$;
  Perform signature process on $M_{Charlie \rightarrow S_{DB}}$: Invoke a function $Signature(M_{Charlie \rightarrow S_{DB}}, d_c, k_{c1}, A_W)$ to return a signature $(r_{c1},s_{c1})$;
  Transfer $M_{Charlie \rightarrow S_{DB}}$ to $S_{DB}$;
Require: Verify the validity of the signature;
  Check whether $T_{now} - T_{Charlie \rightarrow S_{DB}} \leq \tau$ or not;
  Invoke a function $V_{erify}(z'_c,r_{c1},s_{c1})$ and return the verified result;
  if $T_{now} - T_{Charlie \rightarrow S_{DB}} \leq \tau$ then
      if The result is valid then
          Call a chaincode in the name of 'Query File-B' and return $KI_T$ to Charlie;
```

Step 1: Charlie chooses a random integer k_{c1} to generate a query message $M_{Charlie \rightarrow S_{DB}} = h(ID_C \| T_{Charlie \rightarrow S_{DB}} \| KI_T \| A_W)$. Accordingly, Charlie invokes a function $Signature(M_{Charlie \rightarrow S_{DB}}, d_c, k_{c1}, A_W)$ to generate a signature $\{r_{c1},s_{c1}\}$ and submit the parameters $M_{Charlie \rightarrow S_{DB}}(r_{c1},s_{c1})$ to S_{DB}.

Step 2 S_{DB} verifies the timestamp $T_{now} - T_{Charlie \rightarrow S_{DB}} \leq \tau$ to invoke a function $V_{erify}(z'_{c1},r_{c1},s_{c1},A_W)$ in order to confirm the legitimacy of the Charlies' signature. If $(x'_{c1} = r_{c1} \mod n)$ is legal, then Charlie's signature is found to be legitimate to execute a vote-per-share function. This function adds $\{M_{Charlie \rightarrow S_{DB}}, Query_C\}$ to the consortium blockchain where $Query_C = h(r_{c1},s_{c1})$. Lastly, q_d is returned in KI_T. Finally, this phase completes its query process.

Phase IV: Data Transfer In this phase, Ben appeals pvt_{keyB} from Charlie in order to obtain their original data through pvt_{keyB}. The execution flows are as follows:

Step 1: Ben chooses a random integer k_{b2} to compute message request $M_{Ben \rightarrow Charlie} = h(ID_B \| ID_C \| T_{Ben \rightarrow Charlie} \| Hash_{DATA} \| TX_{NO} \| A_W)$. Accordingly, Ben invokes a signature function $Signature(M_{Charlie \rightarrow Ben}, d_c, k_{c2}, A_W)$ to obtain a signature $\{r_{c2}, s_{c2}\}$ for $M_{Charlie \rightarrow Ben}$. Ben uses pub_{keyB} to encrypt the message $M_{Charlie \rightarrow Ben}$ in order to derive $C_{Charlie \rightarrow Ben} = E_p ub_{keyB}(M_{Charlie \rightarrow Ben})$. Lastly, Ben transmits $C_{Charlie \rightarrow Ben}\{r_{c2},s_{c2}\}$ to Charlie.

Step 2: After receiving $C_{Charlie \rightarrow Ben}\{r_{c2}, s_{c2}\}$ from Ben, Charlie decrypts the transmitted message $M_{Charlie \rightarrow Ben} = D_{pvt_{keyB}(C_{Charlie \rightarrow Ben})}$ using pvt_{keyB}. Accordingly, Charlie verifies the timestamp $T_Now - T_{Charlie \rightarrow Ben} \leq \tau$ to invoke a function $V_{erify}(z'_{c2}, r_{c2}, s_{c2}, A_W)$ in order to validate the signature. If $(x'_{c2} = r_{c2} \mod n)$ is legal, then Charlie's signature is found to be legitimate to execute a vote-per-share function. Lastly, Charlie chooses a random integer k_b3 and adds up her session key pvt_{keyB} to the computing message $M_{Ben \rightarrow Charlie} = h(ID_C \| ID_B \| T_{Ben \rightarrow Charlie} \| pvt_{keyB} \| A_W)$. This function invokes $Signature(M_{Ben\rightarrow Charlie},\ d_b,\ k_{b3})$ to encrypt the message transmission $\{r_{b3}, s_{b3}\}$ $M_{Ben\rightarrow Charlie}$ in order to acquire $C_{Ben\rightarrow Charlie} = E_pub_{keyC}(M_{Ben\rightarrow Charlie})$. Lastly, Charlie transmits the system parameters $\{C_{Ben\rightarrow Charlie}, r_{b3}, s_{b3}\}$ to Ben.

Step 3: After receiving the message transmission from Charlie, Ben decrypts the message $M_{Charlie\rightarrow Ben} = D_{pvt_{keyB}}$ using pvt_{keyB}. Accordingly, Ben verifies the timestamp $T_{Now} - T_{Charlie\rightarrow Ben} \leq \tau$ to invoke a function $V_{erify}(z'_{b3},\ r_{b3},\ s_{b3},\ A_W)$ in order to validate the signature. If $(x'_{b3} = r_{b3} \mod n)$ is legal, then Charlie's signature is found to be legitimate to execute a vote-per-share function. Ben acquires the encrypted data of Charlie C_{C1} using KI_T from fabric network and decrypts the transmitted message C_{C1} using Charlie's secret key pvt_{keyC} to obtain $\{M_1 = D_{pvt_{keyC}}(C_{C1}), M_1 = h(ID_B \,\|\, T_1 \,\|\, DT_B) \,\|\, A_W\}$. Lastly, this phase completes its transfer process.

Algorithm 9 Flow Structure of Data Transfer.

```
Require: Submit a transfer data $M_{Ben\rightarrow Charlie}$;
  Compute message request $M_{Ben\rightarrow Charlie} = h(ID_B \| ID_C \| T_{Ben\rightarrow Charlie} \| Hash_{DATA} \| TX_{NO} \| A_W)$;
  Perform signature process on $M_{Ben\rightarrow Charlie}$: Invoke a function $Signature(M_{Charlie\rightarrow Ben}, d_c, k_{c2}, A_W)$ to return a signature $\{r_{c2}, s_{c2}\}$;
  Assign $C_{Charlie\rightarrow Ben} = E_pub_{keyB}(M_{Charlie\rightarrow Ben})$
  Transfer the functional parameters $C_{Charlie\rightarrow Ben}$ and $\{r_{c2}, s_{c2}\}$ to HLF network;
Require: Processed By HLF network [Signcryption];
  After receiving the request, $M_{Charlie\rightarrow Ben} = D_{pvt_{keyB}(C_{Charlie\rightarrow Ben})}$ is validated.
  Check whether $[M_{Charlie\rightarrow Ben} = D_{pvt_{keyB}(C_{Charlie\rightarrow Ben})}]$ satisfies with $T_Now - T_{Charlie\rightarrow Ben} \leq \tau$;
  Initiate a verification process $V_{erify}(z'_{c2}, r_{c2}, s_{c2}, A_W)$ to validate the signature result.
  if $T_Now - T_{Charlie\rightarrow Ben} \leq \tau$ then
      if Result = 'Valid' then
          Charlie chooses a random integer $k_b3$;
          Compute a message request $M_{Ben\rightarrow Charlie} = h(ID_C \| ID_B \| T_{Ben\rightarrow Charlie} \| pvt_{keyB} \| A_W)$;
          Invokes a function call $Signature(M_{Ben\rightarrow Charlie}, d_b, k_{b3})$, and return the message transmission $\{r_{b3}, s_{b3}\}$;
          Transfer the system parameters $\{C_{Ben\rightarrow Charlie}, r_{b3}, s_{b3}\}$ to Ben;
Require: Processed By HLF network [De-signcryption];
  Apply de-Signcryption on $M_{Charlie\rightarrow Ben} = D_{pvt_{keyB}}$ using $pvt_{keyB}$;
  Invoke a function $V_{erify}(z'_{b3}, r_{b3}, s_{b3}, A_W)$ and return its result;
  if $T_{Now} - T_{Charlie\rightarrow Ben} \leq \tau$ then
      if Result = 'Valid' then
          Store $pvt_{keyC}$ and collect the encrypted $DT$ in IPFS network;
          Show $M_1 = D_{pvt_{keyC}}(C_{C1})$
```

Proof of Correctness: Assume $S_j = (r, s_j)$ is a valid signature for the given message N_{msg} over j^{th} period i.e., $(1 \leq j \leq T)$. In consequence, the hash value of the message is set to be $e'_j = H(j, N_{msg}, r) = e_j$ to show its true relationship over $w_j = e'_j + s_j \mod N = e_j + s_j \mod N$ in order to prove the second equality in signature. Additionally, the signature component $s_j = pvt_{key_j}.k - e_j \mod N$ shows its third equality in signature to ensure both privacy and authenticity. Because of robustness over the equality variance i.e., $pvt_{key_j} = a_0^{2^j} \mod N$ and $pub_{key} = a_0^{-1}.G$, the fifth and seventh equality in the signature can be achieved. Considering these variances, the expression of initial private key $pvt_{key_0} = a_0(a_0 \in [1, N-1])$ is recollected, where $a_0 < N$. Therefore, the sixth equality in the signature can be resulted in:

$$\begin{aligned} X = w_j^{2^{-j}}.pub_{key} &= (e_j + s_j \pmod N)^{2^{-j}} \\ pub_{key} &= (pvt_{key_j}.k \pmod N)^{2^{-j}} \\ pub_{key} &= (pvt_{key_j} \pmod N)^{2^{-j}} (m \pmod N)^{2^{-j}} \\ pub_{key} &= (a_{0^{2^j}} \pmod N)^{2^{-j}}.k^{2^{-j}} \\ pub_{key} &= a_0.k^{2^{-j}} \\ pub_{key} &= a_0.k^{2^{-j}}.a_0^{-1}.G = k^{2^{-j}}.G. \end{aligned} \tag{4}$$

From Eq. 4, $(x_2, y_2) = X = k^{2^{-j}}.G = (x_1, y_1$ is obtained. Since the equality in the signature has $r = x_1 \mod N$ and $v = x_2 \mod N$, $v = r$ can be obtained also. As a result, the verifier believes that the construction of the signature $S_j = (r, s_j)$ of N_{msg} is correct. ■

V. Security and Requirement Analysis

This section analyzes the security requirements of the proposed CA-DPPF to examine the transaction results generated by the smart contract and to obtain the corresponding system parameters permitted on the consortium blockchain.

A. Formal Analysis: Using ROM

This model is considered to be more deterministic and publicly accessible with a random uniform distribution function that constantly assigns an appropriate value with deterministic key length using an output field. This assignment makes the query to find an appropriate length of the message in order to return the output value. In general, the random oracle uses the hash function as its idealized form which is adopted by the adversary to derive the preferable hash value. As a result, a simulation is put in practice among the adversary and random oracle to examine the significant abilities of the adversary whereby the provable security is known to break the pseudo-random function. This strategy is regarded as a simulation game to define the adversary queries. At the end of the game, the adversary tries to execute a few pre-deterministic challenges in an effort to succeed in this game. It is worth noting that the pre-deterministic challenge considers knowledge parameters like pub_{key} to solve any underlying problem defined in this scheme. Thus, in case of any success probability of the game, the challenging problem is no more computationally intractable in the real world environment.

1) ***Unforgeability****: Theorem 1. If the ECDLP problem is computationally hard, the proposed CA-DPPF is empirically unforgeable in accordance with the chosen-message attack.*

Proof. Suppose the adversary $\mathcal{A}$ violates the CA-DPPF scheme with key merit ε. As a result, the desired algorithm $\mathcal{D}$ derives a few significant challengers of $\mathcal{A}$ to perform simulation-based operations. They are as follows:

i) *Setup Phase:* $\mathcal{D}$ defines a number of intervals to be T_I and assumes to guess a signature of the J^{th} over the interval $0 \leq J \leq T_I$ devised by $\mathcal{A}$. $\mathcal{D}$ chooses an elliptic curve E_C over a finite prime order $G_F(p)$ and also selects its base point G i.e., on the given order of the elliptic curve over E_C. Additionally, $\mathcal{D}$ chooses the derived hash function $H(.)$ and initial private key x_0 to find $x_0^{-1}.G$. Lastly, $\mathcal{D}$ sends the corresponding output i.e., $x_0^{-1}.G$ to $\mathcal{A}$.

ii) *Query Phase:* The simulated queries are as follows:
   (1) *Hash Query* Initially, this query is simulated to obtain a hash query list L_H of $\mathcal{D}$. It is worth noting that L_H is primarily set to be empty. During the j^{th} interval, $\{N_{msg},e_j\}$ is recorded in L_H. While $\mathcal{A}$ attempts to perform a hash query over a message N_{msg}, $\mathcal{D}$ subsequently operates L_H to determine whether N_{msg} is appeared or not. In case of appearing in L_H, $\mathcal{D}$ replies with e_j to $\mathcal{A}$ immediately. Otherwise, $\mathcal{D}$ chooses $e_j \leftarrow \mathbb{Z}^*_{\ltimes} = \{n = 1,2,3,...,(n-1)\}$ randomly and accordingly records $\{N_{msg}, e_j\}$ in L_H before returning the value of e_j to $\mathcal{A}$.
   (2) *Key Leakage Query* $\mathcal{A}$ contends to process a break-in-function $(j^+)(1 \le J^+ \le T_I$. In case $j^+ \le J$, $\mathcal{D}$ records the process as *failure* to terminate the break-in-function. In case that $j^+ > J$, $\mathcal{D}$ computes the key $pvt_{key_{J+1}}$. Lastly, if $j^+ \ne J$, then $\mathcal{D}$ invokes a key update algorithm to obtain $pvt_{key_{j^+}}$ using $pvt_{key_{j^+}} \leftarrow Update(...,...,Update(pvt_{key_{J+1}}))$ before returning the result $pvt_{key_{j^+}}$ to $\mathcal{A}$.
   (3) *Signature Query* $\mathcal{A}$ processes the signing parameters (j,N_{msg}) where $0 \le j \le J^+$. $\mathcal{D}$ selects $k \in [1,N-1]$ at random to compute $k^{2^{-j}}.G = (x_1,y_1)$, $r = x_1 \pmod N$, $e_j = H(j,N_{msg},r)$, and $s_j = pvt_{key_j} - e_j \pmod N$. Lastly, $\mathcal{D}$ obtain the parameters (r,s_j) to $\mathcal{A}$.

iii) *Forgery Phase* In case of aborting the query phase by $\mathcal{D}$, $\mathcal{A}$ returns an index period j^* i.e., $(0 \le j^* \le j^+)$, message N^*_{msg}, and a valid forged signature (r^*, s^*_j) to verify the property of anonymity against ϵ. When $j^* \ne J$, $\mathcal{D}$ aborts the process. Otherwise, $\mathcal{D}$ returns the value K using $(e_j + s_j).a_0^{-2^j}$. Similarly, the ECDHP can be proven. ■

The above simulation reveals that $\mathcal{D}$ guesses an index period j^* whereby A forges the parameter $j^+ > J$ using key leakage query. As a result, $\mathcal{D}$ obtains $pvt_{key_{j^+}}$ without stopping process. Moreover, $\mathcal{D}$ may guess a relevant index period j^* as it has a success probability $\frac{1}{T}$ to obtain a valid signature to $\mathcal{A}$. Hence, $\epsilon' = \frac{\epsilon}{T} - negl(n)$ is the negligible probability that $\mathcal{D}$ determines the solution of ECDLP successfully.

2) ***Forward Security****: Theorem 2. Since the congruence equation of modulo, N is computationally hard, this improved KA scheme has the property of forward security.*

Proof. In case of obtaining the j^{th} period private key pvt_{key} by $\mathcal{A}$, $\mathcal{A}$ cannot use the j^{th} key to forge the $(j-1)^{th}$ signature. The evidences are as follows:

Initially, $\mathcal{A}$ tries to find the private key $pvt_{key_{(j-1)}}$ of $(j-1)^{th}$ period using $s_{(j-1)} = pvt_{key_{(j-1)}}$. However, this computation demands a proper quadratic congruence of the given modulo N as it is equivalent to the problem of factorization.

Besides, when $\mathcal{A}$ wishes to derive a forge signature (r,s_j) of the given period $(j-1)^{th}$ using $s_{(j-1)} = pvt_{key_{(j-1)}} - e_{(j-1)} \pmod N$, then $\mathcal{A}$ must obtain a value of $pvt_{key_{(j-1)}}$ to produce a falsified signature. However, the above computation reveals that $\mathcal{A}$ cannot derive the $pvt_{key_{(j-1)}}$ of the given period $(j-1)^{th}$ using the j^{th} period private key pvt_{key_j}.

This theorem is proven as follows:

Suppose $\mathcal{A}$ uses a property of forward security to attack this KA scheme. As a result, the algorithm $\mathcal{D}$ constructs the quadratic congruence equation using ∂ as a subroutine function to solve this property. $\mathcal{D}$ applies two functional phases namely chosen-message attack and forgery to gain access to the signature and hash oracle. In consequence, the subroutine ∂ retains in the phase of the chosen-message attack in order to execute the algorithm $\mathcal{D}$. Meanwhile, a random value pvt_{key_j} is chosen by the subroutine to produce a signature for the message N_{msg} at the given index period j^{th}. In this connection, it is assumed that the key pvt_{key} may be disclosed for both periods including the past and the present while performing the execution of *Key Update* phase.

i) The simulation of *Query Phase* is as follows:
   (1) *Signature Query.* We use *Signature* oracle to simulate ∂ which views the signature of the given algorithm $\mathcal{D}$. Let's assume that the signature oracle generates a query message N_{msg} using ∂. As a result, the signature oracle derives a signature (r,s_j) of the given message N_{msg} and accordingly, returns the parameters (r,s_j) to ∂ as a valid response to the signature query. Later, we apply the ∂ view to simulate the signature algorithm which generates the value k to compute $K^{2^{-j}}.G$ using (x_1,y_1) to obtain the value r.
   (2) *Hash Query.* We use ∂ to simulate the *Hash* oracle which returns (j,N_{msg},r) as a query for the given input message N_{msg} in order to obtain the ∂ response.

ii) The simulation of *Forgery Phase* is as follows. Let's assume that in the t^{th} period, the break-in-function obtains the present private key pvt_{key_t} and accordingly, return the value of pvt_{key_t} to ∂. As a result, ∂ derives the forgery signature using $(N_{msg},(r,s_t))$ to obtain the desired output of $\mathcal{D}$.

The success probability of the given algorithm $\mathcal{D}$ shows a similar output as that of its subroutine ∂. However, it has a significant divergence which is as follows: in the simulation, the signature oracle uses the value of $\mathbb{Z}^*_N$ whereas in the real form, the desired output obtains the values from $\mathbb{Z}_N$ instead of $\mathbb{Z}^*_N$. Most importantly, the signature phase randomly selects the value k from $\mathbb{Z}_N$, thus the value obtained from $\mathbb{Z}^*_N$ is negligible. In case $N = pq$, the probability of obtaining the value of k is at most defined as $(p+q)/(pq) = (1/q) + (1/p)$. While q_s is defined as the number of queries submitted to the signature oracle, the success probability of the given algorithm $\mathcal{D}$ is exactly derived as $q_s(1/q) + 1/p$. Hence, the KA scheme achieves the property of forward security. ■

VI. Results and Discussion

The results and discussion evaluate the performance of the proposed CA-DPPF with other existing frameworks [15], [16], [18], [21], [22], [29]–[31] in terms of communication, computation, and storage to determine their effectiveness of decentralization. In addition, a hyper-ledger fabric network using a docker container is utilized to execute the proposed CA-DPPF and other existing frameworks [15], [16], [18], [21], [22], [29]–[31] in order to analyze the performance of file transmission in a fabric network.

A. Analysis I: Cost Efficiency

Communication Cost: In this analysis, two network environments such as 1-Gigabit and 10-Gigabit Ethernet are chosen to examine the transmission speed of the packet. Cost efficiency considers the key agreement phase of the proposed CA-DPPF and other existing frameworks to learn their transmission rounds

TABLE III: Cost Efficiencies of the Proposed CA-DPPF and other Existing Frameworks.

Frameworks Cost ($bits$)	Message Rounds ($\#$)	1-Gigabit (μsec)	10-Gigabit (μsec)	Communication Cost ($bits$)	Computation Cost (ms)	Storage
Proposed CA-DPPF	7	1.837	0.1837	3168	$6T_{E/D}$ + $7T_{HF}$ + $28T_{SM}$ + $16T_{ME}$=$753.28ms$	[448 + 736 = 1184]
Esposito et al. [15]	Applied Systematic Analysis					
Khalid et al. [16]	9	2.368	0.2368	1840	Not Available	[672 + 832 = 1504]
Wang et al. [18]	12	3.679	0.3679	3728	Not Available	[640 + 672 = 1312]
Chen et al. [29]	Applied Systematic Analysis					
Gao et al. [30]	5	1.617	0.1617	1984	$5T_{E/D}$ + $2T_{HF}$ + $4T_{SM}$ = $420.51ms$	896
Liu et al. [31]	10	2.817	0.2817	2784	$4T_{HF}$ + $7T_{SM}$ + $2T_{PA}$ + T_{ME}=$64.83ms$	864
Y Zhang et al. [21]	6	1.731	0.1731	5008	$33T_{HF}$ + $3T_{ME}$ + $6T_{PA}$ = $8.61ms$	960
J Zhang et al. [22]	3	1.271	0.1271	1200	$3T_{HF}$ + $3T_{ME}$ + $2T_{PA}$ = $8.29ms$	992

between the clients and other communication entities via an insecure network. The key agreement phase has the following parameters to define their transmission size: signature and key $\approx 160 bits$, hash operation $\approx 160 bits$, asymmetric encryption/decryption $\approx 1024 bits$, and length of the message transmission i.e., ID, Timestamp, Random Integers $\approx 80 bits$. Considering the data transmission phase, the proposed CA-DPPF has four signatures $[2 \times 160 = 2048 bits]$, two asymmetric encryption/decryption $[2 \times 1024 = 2048 bits]$, two hash functions $[2 \times 160 = 320 bits]$, and two other messages $[2 \times 80 = 160 bits]$. The cumulative cost of the proposed CA-DPPF during data transmission is $3168 bits$, whereas the other existing frameworks are $1840 bits$ [16], $3728 bits$ [18], $1984 bits$ [30], $2784 bits$ [31], $5008 bits$ [21], and $1200 bits$ [22], respectively excluding Esposito et al. [15] and Chen et al. [29] as they do not have any specific message transmission to find or compute the cost of the communication during the process of authorization and validation.

Computation Cost: This analysis includes the key agreement phase of the proposed CA-DPPF and other existing frameworks [15], [16], [18], [21], [22], [29]–[31] to examine their cost efficiencies in terms of the execution time of point addition T_{PA}, scalar multiplication T_{SM}, modular exponentiation T_{ME}, modular inversion T_{MI}, the hash function T_{HF}, and asymmetric encryption/decryption $T_{E/D}$. In order to conduct a practical analysis, a real-time testbed was constructed using MIRACL V7.0 on a dedicated system compatible with Intel (R) Core (TM) i5-1235U, 32GB RAM, and CPU @$4.40GHz$. Initially, the cost efficiencies of the cryptography operations including CA-DPPF and other existing frameworks were executed over 1000 times to compute their average computation cost. The execution time of the operation is as follows: $T_{PA} \approx 0.0811ms$, $T_{SM} \approx 8.8517ms$, $T_{ME} \approx 2.7072ms$, $T_{MI} \approx 0.0471ms$, $T_{HF} \approx 0.0006ms$, and $T_{E/D} \approx 77.02ms$. The cost analysis considers the key agreement phase of the proposed CA-DPPF and other existing frameworks to compute their efficiency factor. Considering the phase, the proposed CA-DPPF has six $T_{E/D}$, seven T_{HF}, twenty eight T_{SM}, and sixteen T_{ME} to determine its efficiency $\approx 753.28ms$. The cost efficiencies of the other existing frameworks are $5T_{E/D} + 2T_{HF} + 4T_{SM} = \approx 420.51ms$ [30], $4T_{HF} + 7T_{SM} + 2T_{PA} + T_{ME} = \approx 64.83ms$ [31], $33T_{HF} + 3T_{ME} + 6T_{PA} = \approx 8.61ms$ [21], and $3T_{HF} + 3T_{ME} + 2T_{PA} = \approx 8.29ms$ [22], respectively.

Storage Cost: This analysis considers the registration phases of the proposed CA-DPPF and other existing frameworks [15], [16], [18], [21], [29]–[31] to determine their storage costs. To derive their efficiencies, the utilized parameters consider with output sizes of the given cryptographic operators and they are as follows: *A point of elliptic curve cryptography* $\approx 160bits$, *user identities* $\approx 128bits$, *random integers* $\approx 160bits$, *timestamp* $\approx 32bits$, *hash* $\approx 160bits$, *cipher-text* $\approx 256bits$, and *key* $\approx 160bits$. In the proposed CA-DPPF, the computing peer uses $\{ID_{E_N}, pub_{key}, pvt_{key}\}$ as the storage parameters to verify its legitimate transaction record in the fabric network. After the successful execution of *Transfer* phase, the computing peer obtains $\{M_1, ID_B, T_1, DT_B, A_W\}$ as the transmitted message to validate the authenticity. While considering the output sizes of the operator, the proposed CA-DPPF requires the storage of $[128 + 160 + 160] = 448bits$ and $[160 + 128 + 32 + 256 + 160] = 736bits$ to verify the transaction of the computing peers whereas the other existing frameworks reserve the storage space of $1504bits$ [16], $1312bits$ [18], $896bits$ [30], $864bits$ [31], $960bits$ [21], and $992bits$ [22], respectively excluding Esposito et al. [15] and Chen et al. [29] as does not hold any specific parameter to validate the transaction proposal submitted by the computing peer.

The comparative analysis shows that the proposed CA-DPPF cannot achieve better cost efficiencies including computation, communication, and storage than other existing frameworks [15], [16], [18], [21], [22], [29]–[31] in some cases, depicted in Table III. However, the proposed CA-DPPF exploits the core features of the KA mechanism to resist security vulnerabilities and also improve the efficiency of the on-chain transaction to offer better data sharing with privacy preservation.

B. Analysis II: Performance Evaluation

This section details the implementation of the proposed CA-DPPF and other existing frameworks which extensively evaluate the performance of the system. Specifically, the experiment analysis considered a few communication metrics such as a committed transaction, average latency, and throughput for each real-time transaction.

1) Case Scenario I: This scenario deployed two levels of consortium blockchain using Hyper-Ledger Fabric (shown in Fig.4) to analyze the cost efficiency which uses smart contracts among three peer computing nodes to seal a transaction in order to prove system feasibility. In a consortium blockchain, the system testbed employed two consensus nodes such as E_{N_A} with CPU: Intel Core i7 4.5GHz Cores: 6, RAM: 16 GB and E_{N_B} with Intel Core i5 4.5GHz Cores: 6, RAM: 16 GB, which employs three computing peers such as Ben, Charlie, and Rose to examine the payment mechanism validated by the KA mechanism in order to examine the trustworthiness using vote-per-share policy. To initiate data transactions and blocks, the application interface (login and payment) utilized user datagram protocol (UDP). Based on the availabilities of the enterprises' computing power, an execution duration was set (i.e., $\approx 4sec$) to the enterprise in an attempt to process the system-defined transactions $\approx 120TPB$.

Also, it is worth noting that the enterprise authority applied a model of POTA (discussed in Section III-D) which employs a genesis block to record a payment store containing data transactions. Each transaction securely links with a chain of blocks using the KA mechanism to verify the peer identities whereby the enterprise uploads the protection of the EHR system using a consortium blockchain to make the execution of smart contracts more efficient and safer. Using the genesis block, E_N records incoming transactions on the blockchain network which is synchronized within each epoch e_t to define the generation of the genesis block. As a result, when N_{um} is the network authorities and δ is the duration time, the execution index is computed by $\tau = e_t/\delta$. Considering this synchronization, E_N obtained a block reward 250DC, and accordingly, it transferred 220DC to 180 different C_{ID} of Ben and converted 27DC into unspent transaction output (UTXO). While initiating the transaction, 0.05DC per transaction was applied.

Similarly, Ben transferred 180DC to Charlie via 120 diverse transactions and earned 67DC as the refunded amount. In the same manner, the crypto-currency can be continuously supplied to the computing peers via a P2P network acting as a distributed ledger to verify the peers' authenticity in order to perform UTXO transactions. Adding to the consortium blockchain, the E_N authority must accept the data block according to the policy made by POTA. As a consequence, the consortium blockchain containing a transaction with a KA verification module and smart contract validates the transaction $T_{x_{pc}}$ generated by the computing nodes. At first, the layer [0] consortium blockchain considered the transaction generated by the computing peer via private blockchain $T_{x_{pc}}$ which was subsequently converted into a new compatible transaction for the consortium blockchain i.e., $T_{x_{CL[0]-CL[1]}}$. This newly generated transaction $T_{x_{pc}}$ was fed to the next computing layer to perform signature verification.

In succession, the layer [1] consortium verified the signature using a smart contract to deliver a verification response to layer [0]. Lastly, layer [0] applied its own smart contract to examine the verification response of $T_{x_{CL[0]-CL[1]}}$. In case of successful verification, layer [0] delivers the verification results of the data transaction to the private blockchain to process further. In order to analyze the performance of the proposed CA-DPPF and other existing frameworks [15], [16], [18], [21], [22], [29]–[31] , the number of transactions per second (TPS) was chosen over a different number of the transaction as input to validate its success rate (i.e., throughput) while the network level was set to be static i.e., 2-layer consortium blockchain. Fig. 5a shows the average number of successfully validated transactions for a 2-layer consortium blockchain network. The examination result reveals that the successful validated transaction decreases when the number of transactions proportionally increases in the given consortium blockchain network. Though there is an adverse effect, the proposed CA-DPPF achieves a better success rate of 98.2% than other existing frameworks.

Additionally, communication metrics such as execution time and latency were considered to evaluate the performance of the proposed CA-DPPF and other existing frameworks [15], [16], [18], [21], [29]–[31]. To do the comparative analysis, in the Hyper-Ledger Fabric-based consortium blockchain, a different number of concurrent transactions (i.e., $\approx$ 1000 to 5000) was considered to compute the execution time and latency of the proposed CA-DPPF and other existing frameworks. Fig. 5b shows the execution time of the validated transaction. The investigation findings show that the proposed CA-DPPF requires around $1.35sec$ to carry out 1000 number of transactions and is comparatively lesser than other existing frameworks $1.65sec$ [15], $1.6sec$ [16], $1.75sec$ [18], $1.55sec$ [29], $1.45sec$ [30], $1.5sec$ [31], $1.7sec$ [21], and $1.8sec$ [22], respectively. Fig. 5c shows the latency of the validated transaction. The study results demonstrate that the proposed CA-DPPF records a lower latency ≈ 0.143 to $0.152sec$ than other existing frameworks 0.152 to $0.175sec$ [15], 0.149 to $0.167sec$ [16], 0.145 to $0.157sec$ [18], 0.146 to $0.161sec$ [29], 0.155 to $0.183sec$ [30], 0.159 to $0.199sec$ [31], and 0.158 to $0.187sec$ [21], and 0.153 to $0.174sec$ [22] respectively since the 2-layer consortium blockchain exhibits a linear increase over the number of concurrent transactions.

VII. Conclusion

This paper contributed an overview of the application of blockchain and smart contracts for a medical center that has sets of patients/doctors, managers, and networks. It is highly preferred by medical enterprises such as pharmaceutical companies for improving security and privacy. It enforces blockchain technology to decentralize control management and solve the issues of system compatibility and interoperability. It also imparts the advantages of a private blockchain over the public blockchain exclusively for activities inside the organization. The advantages of the blockchain-based system over the traditional centralized system include being faster at increasing the performance efficiencies of transactions. Thus, this paper presented a cloud-assisted decentralized privacy-preserving framework, which uses blockchain technology and acquires adequate computing resources to improve data integrity, energy consumption, security, and privacy. Moreover, the formal and informal analysis proves that the proposed CA-DPPF achieves better security efficiency than the other existing frameworks to conduct protection analysis over the chain of ledgers.

Suitable time consumption increases the transaction process exponentially when two or more entities do not belong to the same region.

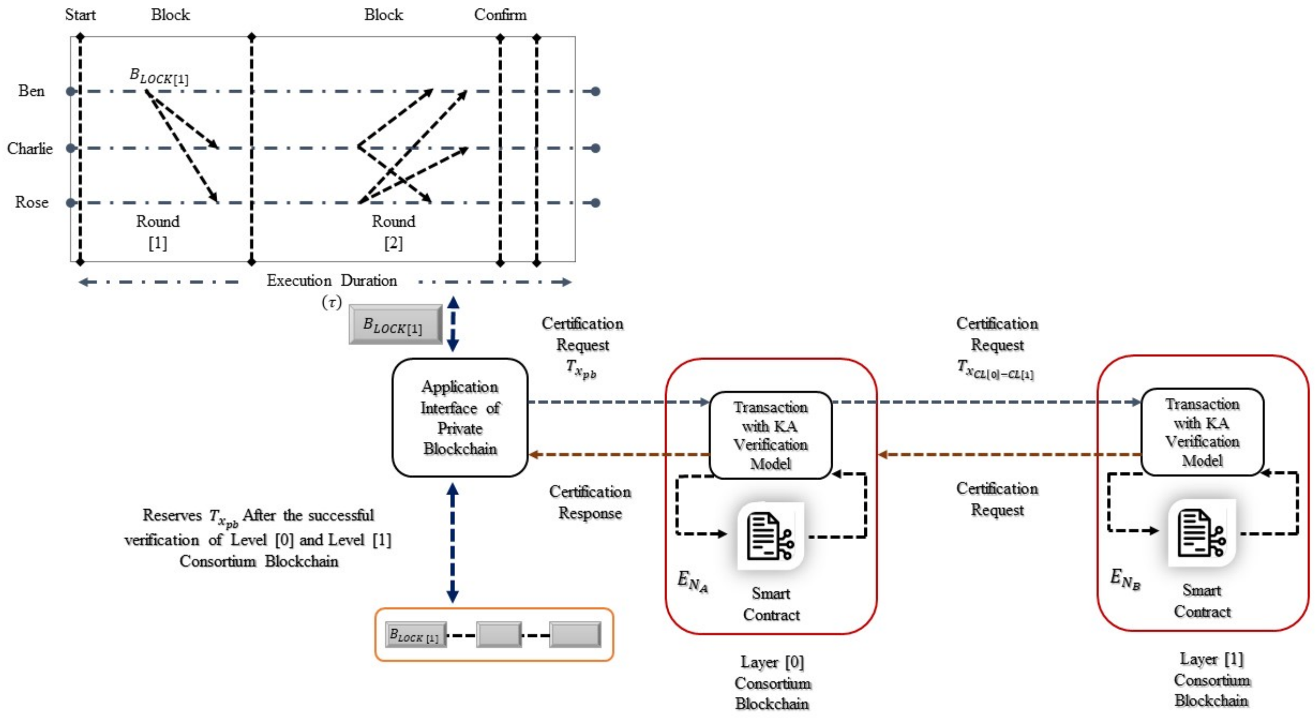
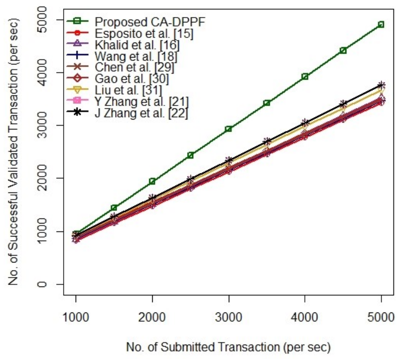
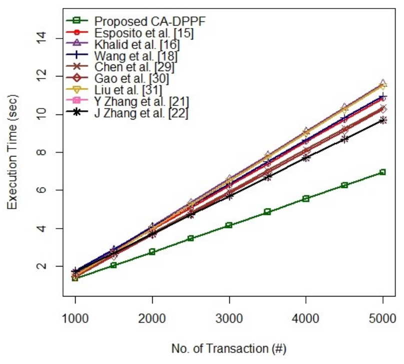

Start
Block
Block
Confirm
Ben
Charlie
Rose
$B_{LOCK[1]}$
Round [1]
Round [2]
Execution Duration (τ)
Certification Request $T_{x_{pb}}$
Application Interface of Private Blockchain
Transaction with KA Verification Model
Certification Request $T_{x_{CL[0]-CL[1]}}$
Certification Response
Certification Request
Reserves $T_{x_{pb}}$ After the successful verification of Level [0] and Level [1] Consortium Blockchain
E_{N_A}
E_{N_B}
Smart Contract
Layer [0] Consortium Blockchain
Layer [1] Consortium Blockchain

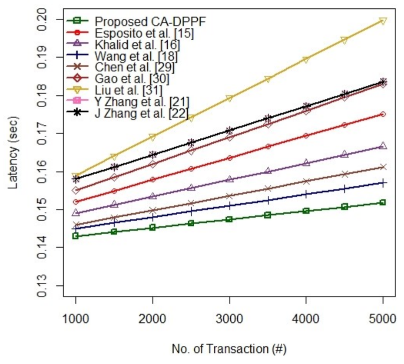
Fig. 4: Transaction Scenario of Two-Layer Consortium Blockchain.



(a) No. of validated transactions $(\#)$



(b) Execution Time (sec)



(c) Latency (sec)

Fig. 5: Analyzing the cost efficiency using two levels of consortium blockchain.

This eliminates the problems caused by network behavior in terms of forcible restrictions. The blockchain-based approach with the help of smart contracts achieves objectives such as shorter transaction delays, secure transactions, secure storage, and easy record accessibility to speed up transactions, minimize power consumption, and maximize data privacy. Finally, the performance analysis demonstrates that the proposed CA-DPPF can fulfill the desired objectives of the healthcare applications such as maximum committed transaction, minimum latency, and maximum throughput to examine the session information supplied by the service agents. In the future, this work will be extended to optimize the consensus mechanism of the fabric network whereby the peer node endorses the legitimacy of the transaction contents to enhance processing capacity and transparency with a faster transaction speed using smart contracts.

References

[1] V. Gatteschi, F. Lamberti, C. Demartini, C. Pranteda, and V. Santamarıa, "Blockchain and smart contracts for insurance: Is the technology mature enough?" *Future internet*, vol. 10, no. 2, p. 20, 2018.

[2] J. Grover, "Security of vehicular ad hoc networks using blockchain: A comprehensive review," *Vehicular Communications*, vol. 34, p. 100 458, 2022.

[3] M. Jiang and X. Qin, "Distributed ledger technologies in vehicular mobile edge computing: A survey," *Complex & Intelligent Systems*, vol. 8, no. 5, pp. 4403–4419, 2022.

[4] W. Hao, J. Zeng, X. Dai, *et al.*, "Towards a trust-enhanced blockchain p2p topology for enabling fast and reliable broadcast," *IEEE Transactions on Network and Service Management*, vol. 17, no. 2, pp. 904–917, 2020.

[5] W. Al-Saqaf and N. Seidler, "Blockchain technology for social impact: Opportunities and challenges ahead," *Journal of Cyber Policy*, vol. 2, no. 3, pp. 338–354, 2017.

[6] I. A. Omar, R. Jayaraman, M. S. Debe, K. Salah, I. Yaqoob, and M. Omar, "Automating procurement contracts in the healthcare supply chain using

blockchain smart contracts," *IEEE Access*, vol. 9, pp. 37 397–37 409, 2021.

[7] E. Borgia, "The internet of things vision: Key features, applications and open issues," *Computer Communications*, vol. 54, pp. 1–31, 2014.

[8] J. Yang, J. Wen, B. Jiang, and H. Wang, "Blockchain-based sharing and tamper-proof framework of big data networking," *IEEE Network*, vol. 34, no. 4, pp. 62–67, 2020.

[9] B. D. Deebak, F. Al-Turjman, M. Aloqaily, and O. Alfandi, "An authentic-based privacy preservation protocol for smart e-healthcare systems in iot," *IEEE Access*, vol. 7, pp. 135 632–135 649, 2019.

[10] P. P. Ray, D. Dash, K. Salah, and N. Kumar, "Blockchain for iot-based healthcare: Background, consensus, platforms, and use cases," *IEEE Systems Journal*, vol. 15, no. 1, pp. 85–94, 2020.

[11] D. Chattaraj, B. Bera, A. K. Das, S. Saha, P. Lorenz, and Y. Park, "Block-clap: Blockchain-assisted certificateless key agreement protocol for internet of vehicles in smart transportation," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 8092–8107, 2021.

[12] F. Al-Turjman, B. D. Deebak, and L. Mostarda, "Energy aware resource allocation in multi-hop multimedia routing via the smart edge device," *IEEE Access*, vol. 7, pp. 151 203–151 214, 2019.

[13] H. Kaur, M. A. Alam, R. Jameel, A. K. Mourya, and V. Chang, "A proposed solution and future direction for blockchain-based heterogeneous medicare data in cloud environment," *Journal of medical systems*, vol. 42, pp. 1–11, 2018.

[14] G. Nagasubramanian, R. K. Sakthivel, R. Patan, A. H. Gandomi, M. Sankayya, and B. Balusamy, "Securing e-health records using keyless signature infrastructure blockchain technology in the cloud," *Neural Computing and Applications*, vol. 32, pp. 639–647, 2020.

[15] C. Esposito, M. Ficco, and B. B. Gupta, "Blockchain-based authentication and authorization for smart city applications," *Information Processing & Management*, vol. 58, no. 2, p. 102 468, 2021.

[16] U. Khalid, M. Asim, T. Baker, P. C. Hung, M. A. Tariq, and L. Rafferty, "A decentralized lightweight blockchain-based authentication mechanism for iot systems," *Cluster Computing*, vol. 23, no. 3, pp. 2067–2087, 2020.

[17] A. A.-N. Patwary, A. Fu, S. K. Battula, R. K. Naha, S. Garg, and A. Mahanti, "Fogauthchain: A secure location-based authentication scheme in fog computing environments using blockchain," *Computer Communications*, vol. 162, pp. 212–224, 2020.

[18] Y. Wang, A. Zhang, P. Zhang, and H. Wang, "Cloud-assisted ehr sharing with security and privacy preservation via consortium blockchain," *Ieee Access*, vol. 7, pp. 136 704–136 719, 2019.

[19] S. Karumba, S. S. Kanhere, R. Jurdak, and S. Sethuvenkatraman, "Harb: A hypergraph-based adaptive consortium blockchain for decentralized energy trading," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14 216–14 227, 2020.

[20] E. Samir, H. Wu, M. Azab, C. Xin, and Q. Zhang, "Dt-ssim: A decentralized trustworthy self-sovereign identity management framework," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 7972–7988, 2021.

[21] Y. Zhang, B. Li, J. Wu, B. Liu, R. Chen, and J. Chang, "Efficient and privacy-preserving blockchain-based multifactor device authentication protocol for cross-domain iiot," *IEEE Internet of Things Journal*, vol. 9, no. 22, pp. 22 501–22 515, 2022.

[22] J. Zhang, Y. Jiang, J. Cui, D. He, I. Bolodurina, and H. Zhong, "DBCPA: Dual blockchain-assisted conditional privacy-preserving authentication framework and protocol for vehicular ad hoc networks," *IEEE Transactions on Mobile Computing*, pp. 1–15, 2022. DOI: 10.1109/tmc.2022.3230853. [Online]. Available: https://doi.org/10.1109/tmc.2022.3230853.

[23] F. Cai, N. Zhu, J. He, P. Mu, W. Li, and Y. Yu, "Survey of access control models and technologies for cloud computing," *Cluster Computing*, vol. 22, pp. 6111–6122, 2019.

[24] S. Rouhani and R. Deters, "Blockchain based access control systems: State of the art and challenges," in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 423–428.

[25] K. Renuka, S. Kumari, and X. Li, "Design of a secure three-factor authentication scheme for smart healthcare," *Journal of medical systems*, vol. 43, pp. 1–12, 2019.

[26] P. Pandey and R. Litoriya, "Securing e-health networks from counterfeit medicine penetration using blockchain," *Wireless Personal Communications*, vol. 117, pp. 7–25, 2021.

[27] C. C. Agbo and Q. H. Mahmoud, "Comparison of blockchain frameworks for healthcare applications," *Internet Technology Letters*, vol. 2, no. 5, e122, 2019.

[28] S. Tanwar, K. Parekh, and R. Evans, "Blockchain-based electronic healthcare record system for healthcare 4.0 applications," *Journal of Information Security and Applications*, vol. 50, p. 102 407, 2020.

[29] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-based medical records secure storage and medical service framework," *Journal of medical systems*, vol. 43, pp. 1–9, 2019.

[30] S. Gao, Q. Su, R. Zhang, J. Zhu, Z. Sui, and J. Wang, "A privacy-preserving identity authentication scheme based on the blockchain," *Security and Communication Networks*, vol. 2021, pp. 1–10, 2021.

[31] S. Liu, Y. Chai, L. Hui, and W. Wu, "Blockchain-based anonymous authentication in edge computing environment," *Electronics*, vol. 12, no. 1, p. 219, 2023.

[32] S. Roy, A. K. Das, S. Chatterjee, N. Kumar, S. Chattopadhyay, and J. J. Rodrigues, "Provably secure fine-grained data access control over multiple cloud servers in mobile cloud computing based healthcare applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 1, pp. 457–468, 2018.

[33] J. Srinivas, A. K. Das, N. Kumar, and J. J. Rodrigues, "Cloud centric authentication for wearable healthcare monitoring system," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 5, pp. 942–956, 2018.

[34] M. Wazid, A. K. Das, and S. Shetty, "Bsfr-sh: Blockchain-enabled security framework against ransomware attacks for smart healthcare," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 1, pp. 18–28, 2022.

[35] S. Itoo, A. A. Khan, V. Kumar, A. Alkhayyat, M. Ahmad, and J. Srinivas, "Ckmib: Construction of key agreement protocol for cloud medical infrastructure using blockchain," *IEEE Access*, vol. 10, pp. 67 787–67 801, 2022.

[36] A. K. Sahu, S. Sharma, and D. Puthal, "Lightweight multi-party authentication and key agreement protocol in iot-based e-healthcare service," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 2s, pp. 1–20, 2021.

[37] R. Beck, C. Müller-Bloch, and J. L. King, "Governance in the blockchain economy: A framework and research agenda," *Journal of the Association for Information Systems*, vol. 19, no. 10, p. 1, 2018.

[38] M. Janssen, V. Weerakkody, E. Ismagilova, U. Sivarajah, and Z. Irani, "A framework for analysing blockchain technology adoption: Integrating institutional, market and technical factors," *International Journal of Information Management*, vol. 50, pp. 302–309, 2020.

[39] L. Feng, H. Zhang, Y. Chen, and L. Lou, "Scalable dynamic multi-agent practical byzantine fault-tolerant consensus in permissioned blockchain," *Applied Sciences*, vol. 8, no. 10, p. 1919, 2018.

[40] T. A. Alghamdi, I. Ali, N. Javaid, and M. Shafiq, "Secure service provisioning scheme for lightweight iot devices with a fair payment system and an incentive mechanism based on blockchain," *IEEE Access*, vol. 8, pp. 1048–1061, 2019.

[41] L. Zavolokina, R. Ziolkowski, I. Bauer, and G. Schwabe, "Management, governance and value creation in a blockchain consortium," *MIS Quarterly Executive*, vol. 19, no. 1, pp. 1–17, 2020.

[42] K. Abbas, M. Afaq, T. Ahmed Khan, and W.-C. Song, "A blockchain and machine learning-based drug supply chain management and recommendation system for smart pharmaceutical industry," *Electronics*, vol. 9, no. 5, p. 852, 2020.