GDPR-Compliant e-KYC using Redactable Blockchain and Zero Knowledge Proof

Piphattra Sreekongpan, Supawit Nakprame, Surabordin Sungchai
School of ICT, Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani, Thailand
sr_pipattra@outlook.com, 6422770287@g.siit.tu.ac.th, 6422782514@.siit.tu.ac.th

***Abstract*—** The increasing reliance on electronic Know Your Customer (e-KYC) systems by financial institutions necessitates compliance with stringent privacy regulations, such as the General Data Protection Regulation (GDPR), to safeguard personal data. However, traditional blockchain-based solutions struggle to accommodate privacy regulations like GDPR due to their immutability, especially in the context of a user's right to request the deletion of personal data. This paper introduces a blockchain-based framework that integrates redactable and privacy-aware techniques, such as chameleon hashing and zero-knowledge proofs (ZKPs), to provide a GDPR-compliant e-KYC solution that balances accountability, transparency, and user privacy. The framework not only enhances accountability and privacy but also addresses the scalability and performance issues typically associated with traditional blockchain-based solutions. The proposed framework demonstrates improved performance metrics compared to conventional blockchains, highlighting the potential for practical implementation in real-world e-KYC and data privacy applications.

Keywords—eKYC, GDPR, redactable blockchain, zero knowledge proofs

I. INTRODUCTION

The rapid growth of e-commerce has heightened the demand for secure, reliable, and efficient customer verification mechanisms. Among these, the Know Your Customer (KYC) process has emerged as a widely adopted approach to verify customer identities, preventing fraud, money laundering, and other financial crimes. By collecting and verifying personal information, such as government-issued IDs and proof of address, businesses can confirm the legitimacy of customers, thereby mitigating risks associated with online transactions. As digital platforms continue to expand, KYC has become an essential component in maintaining trust, security, and regulatory compliance across the e-commerce industry.

However, traditional KYC processes often rely on manual, paper-based verification systems that are not only time-consuming and error-prone but also susceptible to data breaches. The advent of blockchain technology has introduced a paradigm shift, enabling decentralized and secure solutions for managing sensitive customer information. Despite its advantages, privacy concerns regarding the storage and sharing of personal data remain a pressing challenge.

Electronic Know Your Customer (e-KYC) has gained widespread adoption among financial institutions (FIs) and banks worldwide, facilitating backend customer authentication without the direct exchange of credentials between parties. Modern e-KYC systems employ cryptographic-based access control solutions and advanced authentication mechanisms, such as two-factor authentication (2FA) and multi-factor authentication (MFA). Additionally, secure data exchanges between FIs are achieved through encryption and adherence to strict authorization policies. Blockchain technology further enhances e-KYC by improving security, traceability, and efficiency in banking applications. Its decentralized and immutable nature offers a secure, cost-effective way to streamline KYC processes, ensuring customer privacy, reducing fraud, and complying with regulatory requirements. This makes blockchain an ideal solution for digital identity verification across the e-commerce and financial sectors, reducing dependence on centralized trusted third parties.

Despite these advancements, existing e-KYC systems face several unresolved challenges. First, scalable and anonymous authentication is critical, particularly when processing e-KYC requests involving a large number of customers and non-host FIs. Current e-KYC systems rely heavily on centralized authentication mechanisms like 2FA and MFA, managed by the core systems of individual FIs. These centralized approaches incur high communication costs and create bottlenecks during peak request volumes, especially in scenarios requiring large-scale e-KYC validations. For example, when multiple non-host FIs simultaneously request e-KYC validation for loan applications, centralized authentication systems often face delays due to network congestion and computation overload on the host FI.

Second, dynamic data portability is a pressing requirement in real-world applications. Financial institutions frequently need to share customer data securely and efficiently during processes like loan approvals, financial audits, or cross-border transactions. However, existing e-KYC systems struggle to provide flexible and secure mechanisms for dynamic

authentication and data transfer between multiple FIs while ensuring compliance with regulatory frameworks.

Third, compliance with privacy regulations such as GDPR [1] poses a significant challenge. Blockchain's immutable nature conflicts with the "right to be forgotten" clause, making it difficult to erase or redact personal data upon customer request. Additionally, ensuring that e-KYC verification requests remain confidential and occur only with explicit customer consent is a persistent issue in blockchain-based e-KYC systems.

To address these challenges, we propose a novel secure and flexible, GDPR-compliant e-KYC scheme leveraging a redactable blockchain architecture. Our solution integrates advanced techniques to ensure secure, scalable, and efficient customer verification. First, we introduce an optimized authentication mechanism that combines Zero-Knowledge Rollup (ZK-Rollup) and a novel sharded proof verification strategy. This mechanism enables transparent validation of customer credentials while efficiently handling large volumes of e-KYC requests. e-KYC transactions are grouped based on proximity to distributed verification services, batched into a single succinct Zero-Knowledge Proof off-chain, and then submitted to the blockchain. This approach significantly reduces on-chain computation and storage requirements, improving throughput and scalability.

Second, to facilitate secure and dynamic data portability, our scheme incorporates adaptive proxy re-encryption. This mechanism enables encrypted data transfer between host and requesting FIs without exposing plaintext information. The system dynamically adjusts to varying authentication requirements and ensures secure access to customer data for critical operations, such as loan approvals and regulatory audits.

Third, to ensure compliance with privacy regulations, we propose an optimized redactable blockchain protocol that leverages the Chameleon Hash algorithm for controlled data redaction. Our approach integrates off-chain redaction execution to improve scalability and efficiency, while the security of the redaction process is ensured through our proposed dynamic key rotation mechanism. This design guarantees privacy compliance, data integrity, and adaptability in managing sensitive customer information. To the best of our knowledge, we provide the first attempt in addressing the "right to be forgotten" issue of GDPR in blockchain-based e-KYC.

In summary, the key contributions of our proposed system include:

1. **Scalable and Efficient Authentication**: The integration of ZK-Rollup and sharded proof verification facilitates the handling of large-scale e-KYC requests with minimal on-chain computational overhead, ensuring high throughput and efficiency. Off-chain proof generation and batching significantly reduce computational and storage costs, making the system more economical and scalable for real-world applications. +Sharded BC
2. **Dynamic and Secure Data Portability**: The system incorporates adaptive proxy re-encryption to enable secure and flexible data sharing between host and requesting financial institutions (FIs). This ensures that critical operations like loan approvals and regulatory audits are supported seamlessly.
3. **Secure and Optimized Redaction process**: By leveraging the Chameleon Hash algorithm, combined with off-chain redaction execution, the protocol allows controlled modifications to stored data without compromising blockchain integrity. Additionally, the implementation of dynamic key rotation enhances the security of the redaction process and ensures long-term compliance with privacy regulations, including GDPR.

II. RELATED WORK

This section discusses works entailing e-KYC systems and literature addressing the redactable blockchain.

Kumar et al. [8] proposed a blockchain-based approach to decentralize the storage of personal data in the KYC process, addressing inefficiencies and operational redundancies in traditional methods. Their scheme focuses on data sovereignty by encrypting users' Personally Identifiable Information (PII), which is stored in a QR code format, ensuring a two-layer security mechanism. Access to this data is granted only when explicitly authorized by the customer. Additionally, the system emphasizes enhanced user privacy and cost reduction through decentralized coordination among banks, regulators, and stakeholders. However, the paper does not explore mechanisms to facilitate GDPR compliance or adaptability in cases of data modification.

Mamun et al. [9] proposed a secure and transparent KYC document verification system for banking using InterPlanetary File System (IPFS) and blockchain technology. Their approach streamlines the KYC process by enabling customers to complete the process once at a single bank, generating a hash value via the IPFS network and sharing it securely through blockchain. This allows other banks or financial organizations to retrieve and store the customer's data with the customer's consent, eliminating redundant KYC processes. The system aims to save time, reduce costs, and enhance security while facilitating account creation at multiple institutions. However, the proposed solution does not address regulatory compliance frameworks, such as GDPR, or the ability to modify stored data while maintaining integrity.

Fugkeaw [2] introduced e blockchain-based e-KYC scheme, known as e-KYC TrustBlock, which leverages ciphertext-policy attribute-based encryption (CP-ABE) combined with client consent enforcement to ensure trust, security, and compliance with privacy standards. Furthermore, the scheme incorporates attribute-based encryption to facilitate

privacy-preserving and fine-grained access control for sensitive transactions stored on the blockchain. A smart contract is also developed to create and enforce digital consents signed by customers, with these consents systematically recorded on the blockchain. The author also presented a policy update algorithm to enable efficient re-encryption, utilizing a simplified policy tree structure. However, this paper did not focus on high volume of -KYC transactions.

Suga [3] introduced the e-KYC-e model, leveraging the ERC735 claim-style credentials in the Ethereum blockchain to enhance digital identity verification. The model utilizes decentralized identifiers (DIDs) and self-sovereign identity (SSI) concepts, enabling individuals to control their digital identities. Through a metaphor of "cotton candy," the paper illustrates how multiple attributes (claims) can be linked to a specific identifier, creating a comprehensive and verifiable digital identity. The proposed system demonstrates its applicability by providing digital proof of official certificates, such as enrollment records, within a blockchain framework. However, the paper focuses more on the management of digital identities and claims rather than addressing regulatory compliance, such as GDPR, or the flexibility needed for modifying claims within the blockchain environment.

Patil and Sangeetha [7] proposed a blockchain-based decentralized KYC verification framework for banks, leveraging the Ethereum blockchain to enhance the efficiency, security, and transparency of the KYC process. The framework enables all banks within the blockchain network to verify the legitimacy of customer-provided data through a voting mechanism. The KYC status of a customer is stored on the blockchain based on the consensus achieved through this process. A unique aspect of the framework is its provision for banks to vote on the behavior of other banks, allowing the removal of entities that tamper with KYC data. While this approach addresses several limitations of traditional manual KYC processes, it does not specifically focus on regulatory compliance, such as GDPR, or support flexible data modification in response to evolving customer information.

Schlatt et al. [11] proposed a framework for digital KYC processes leveraging blockchain-based self-sovereign identity (SSI) to address inefficiencies and privacy concerns in traditional KYC methods. The framework employs a design science research approach to integrate SSI, enabling customers to maintain control over their personal data while ensuring compliance with data protection regulations. The study derives design principles that theorize blockchain’s role in facilitating SSI, offering a privacy-preserving and efficient solution to KYC challenges.

Takaragi et al. [13] proposed a privacy-preserving eKYC framework tailored for Central Bank Digital Currencies (CBDCs), integrating delegatable anonymous credentials (DAC) and zero-knowledge range proofs (ZKRP). The framework allows customers to selectively disclose parts of the issuer's delegation chain and document details while retaining control over their privacy. By leveraging zero-knowledge proofs, the system ensures that sensitive information, such as time stamps and ID registration validity, can be verified without revealing their content. This approach enables privacy-enhanced public key infrastructure (PKI) and supports self-sovereign identity management, contributing to a sustainable financial system. However, while the study addresses privacy and revocation challenges effectively, it does not focus on regulatory compliance frameworks like GDPR or the flexibility to edit stored data while maintaining integrity.

Jaber et al. [4] proposed a conceptual framework for integrating blockchain technology into the banking sector's e-KYC systems. The framework emphasizes enhancing security, operational efficiency, and real-time verification through features such as digital documentation, biometric verification, electronic consent, and blockchain integration. The study highlights the potential of blockchain to streamline lending processes, reduce counterparty risk, and improve return on investment while addressing challenges like regulatory compliance, data privacy, scalability, and interoperability.

Bhatia et al. [10] proposed a cost-efficient blockchain-based e-KYC platform that leverages biometric verification to address the limitations of traditional and digitized KYC processes. The platform allows users to complete video-based KYC verification after making an ether payment, upon which they receive a unique KYC key. This key enables financial institutions (FIs) to verify user authentication directly on the blockchain without requiring repeated submission of documents, ensuring privacy and reducing redundancy. By leveraging blockchain's decentralized, transparent, and immutable properties, the proposed platform enhances security, minimizes operational costs, and streamlines the KYC process.

Patkar et al. [12] proposed a privacy-preserving and trustworthy e-KYC system utilizing blockchain technology to address the inefficiencies and privacy concerns associated with traditional centralized KYC processes. The system leverages a decentralized blockchain network to securely store and manage customer identity data, eliminating the need for repetitive verification across multiple financial institutions. By employing cryptographic techniques, the proposed framework enhances data security, ensures transparency in data sharing with customer consent, and reduces operational costs. This approach optimizes the KYC verification process, fostering trust between customers and financial institutions. However, the study does not explore scalability challenges or integration with existing Aadhaar-based authentication systems.

Recently, Ahmed et al. [6] introduced a novel approach to enhance the security and efficiency of e-KYC systems by integrating blockchain technology, Web 3.0, and quantum computing. The proposed framework addresses limitations in current e-KYC solutions, such as high key management costs and reliance on traditional encryption, by employing quantum encryption key methodologies to ensure stronger data protection and confidentiality. This approach enables faster and more reliable customer verification, improving productivity and mitigating the vulnerabilities of conventional encryption techniques. The study highlights how the combined potential of Web 3.0 and quantum computing can revolutionize the e-KYC landscape, offering secure and private solutions.

Nevertheless, all the works mentioned above highlight significant advancements in e-KYC systems; however, they do not fully address the inherent conflict between blockchain's immutability and regulatory requirements such as the General Data Protection Regulation (GDPR). This conflict poses challenges in managing erroneous or sensitive data. Recent advancements in redactable blockchain models offer a promising solution, striving to balance data integrity with compliance, privacy, and usability.

Yeh et al. [14] proposed a GDPR-compliant personal health record (PHR) sharing mechanism that leverages redactable blockchain and revocable IPFS to address privacy and data integrity concerns in the eHealth environment. Their approach integrates redactable blockchain technology with a revocable IPFS mechanism, enabling selective removal of records and files while ensuring compliance with GDPR's right to erasure. Additionally, the study introduces an enhanced proxy re-encryption scheme to simplify access control for healthcare professionals without the complexities of group key management. By offering a secure, efficient, and compliant solution for PHR sharing, this work presents a significant advancement in the ability to manage personal health data on blockchain platforms

Zhou et al. [15] introduced a novel redactable blockchain approach that utilizes trapdoor-hash to address challenges posed by the immutability of traditional blockchain systems. The study focuses on enabling efficient transaction editing, deletion, and smart contract repair while maintaining the integrity and security of the blockchain. Their solution provides fine-grained control over modification authority, allowing users to amend their own transactions and regulators to remove erroneous transactions. The approach has been successfully integrated with Hyperledger Fabric, demonstrating low overhead and greater flexibility in permission control compared to both immutable and other redactable blockchain systems.

Dong et al. [18] proposed a decentralized consortium blockchain system prototype that incorporates redactability and access control to address privacy concerns and ensure secure data sharing. The system leverages chameleon hash and multi-authority attribute-based encryption to enable authorized individuals to remove or replace undesirable content while limiting data visibility to authorized users. This approach allows for the implementation of a regulated consortium blockchain that combines the benefits of secure access control and efficient redactability. The study demonstrates the feasibility of integrating these mechanisms within a blockchain system, highlighting its potential for real-world applications. However, the work primarily focuses on the technical feasibility and does not extensively explore scalability or broader adoption challenges in diverse industry contexts.

Zhang et al. [19] introduced a dynamic trust-based redactable blockchain that addresses the challenges of blockchain's immutability and the potential for misuse in storage. Their proposed solution incorporates a dynamic trust evaluation model that considers multiple factors, including user behavior, to assess the reliability of participants. The model combines dynamic proactive secret sharing (DPSS), chameleon hash (CH), and digital signature (DS) to ensure full-process security with pre-modification evaluation, modification privilege restrictions, and post-modification traceability. The authors demonstrate the effectiveness of their approach by applying it to a consortium blockchain, highlighting its improved security and performance over existing systems. This work significantly enhances blockchain's adaptability, providing a controlled and secure mechanism for updates and traceability in real-world applications

Wang et al. [20] proposed a secure and trusted copyright protection method for educational data using a redactable blockchain to address challenges in managing and protecting educational multimedia content. Their approach incorporates a decentralized chameleon hash function for efficient and trusted copyright maintenance, allowing for rapid modification of infringing content. Additionally, they introduce a Proof of Behavior (PoB) consensus mechanism based on a Bayesian network (BN) to accelerate consensus processing by predicting user behavior and selecting highly trusted nodes for participation. The proposed solution improves system scalability and security while reducing storage requirements by 5%-10% compared to other blockchain-based methods. This work provides a promising framework for enhancing copyright protection in the educational sector while maintaining performance and privacy.

Xu et al. [21] introduced a privacy-preserving and redactable healthcare blockchain system (PRHBS) to address the challenges of data confidentiality and flexibility in key distribution within healthcare applications. Their approach incorporates fine-grained block-level data redaction and secure data sharing through flexible key distribution mechanisms, aligning with the General Data Protection Regulation (GDPR) requirements, such as the "right to be forgotten." The proposed system utilizes a trapdoor-based chameleon hash function, attribute-based encryption, and puncturable encryption to enhance security while ensuring data confidentiality in multi-user settings. The study provides formal security models and demonstrates the effectiveness of PRHBS through a comprehensive comparison, showcasing its superior functionality and performance compared to existing solutions. This work significantly contributes to the development of secure, flexible, and GDPR-compliant blockchain systems in healthcare.

Guo et al. [17] proposed a decentralized approach to managing Electronic Health Records (EHR) using a redactable blockchain and revocable IPFS, addressing the challenges posed by blockchain's immutability in the eHealth sector. The authors introduced a novel attribute-based chameleon hash function that enables secure and efficient transaction-level redactions of EHRs, mitigating privacy and security concerns. This system, implemented using the Charm cryptographic library and a revocable IPFS scheme, supports rapid redactions in seconds. The study demonstrates the practical efficiency of the proposed system through extensive performance experiments on both blockchain and IPFS, showcasing its viability for secure and dynamic EHR management in the context of the Internet of Health Things (IoHT).

Recently, Heo et al. [16] introduced a privacy-preserving redactable blockchain solution that addresses identity tracing challenges within decentralized systems. The approach leverages meta-transactions and zk-SNARKs to enhance anonymity by generating one-time cryptographic keys for each transaction, preventing identity tracing. Additionally, zk-SNARKs are used to hide cryptographic keys and signatures, ensuring greater privacy. The system maintains concise modification histories and enables rapid verification, with modification times around 10 milliseconds even for transactions with multiple changes. The authors also propose a redaction fee scheme to incentivize transaction owners to remove data rather than modify it, thus minimizing performance overhead and ensuring efficient ledger management.

III. PRELIMINARIES

This section provides an overview of the foundational technologies and concepts utilized in our framework, which combines e-KYC, GDPR, blockchain, chameleon hashing, and zero-knowledge proofs. We also discuss related works, highlighting the existing literature in these fields.

ZERO-KNOWLEDGE PROOF

In the context of redactable blockchain, Zero-Knowledge Proof (ZKP) is a cryptographic method that allows a party to demonstrate the truth of a statement without revealing any underlying data. ZKPs are pivotal for enhancing privacy, particularly when sensitive information must be validated without exposure. In a redactable blockchain, ZKPs allow for the verification of transactions while ensuring that sensitive data remains hidden. This is especially important in e-KYC (electronic Know Your Customer) processes, where user data must be validated for compliance without compromising privacy. By incorporating ZKPs, our framework ensures that personal information stays private while still being verifiable, aligning with the requirements of GDPR. This technology is essential for creating trust in blockchain-based identity verification systems, offering a balance between transparency and privacy. It ensures that only the necessary elements of a transaction are revealed, protecting users' personal information while providing assurance to external parties about the transaction's legitimacy.

Chameleon Hash

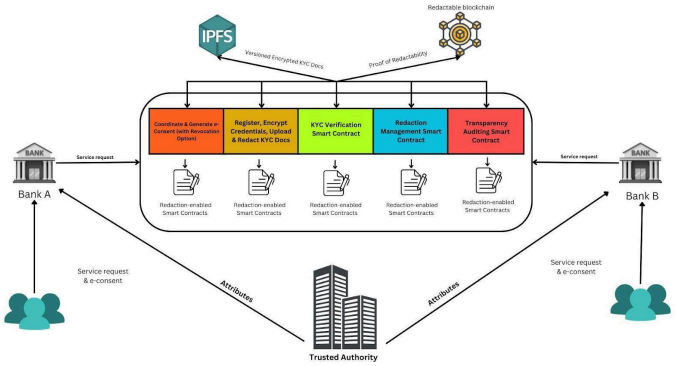

Fig. 1. System model

IV. OUR PROPOSED SCHEME

A. SYSTEM MODEL

We propose a GDPR-compliant e-KYC system using a redactable blockchain Figure 1 illustrates the system overview of our proposed scheme.

The system model consists of the following entities:

1. **Attribute Authority:** The authority in this system is responsible for generating the public parameters (PK) and the master secret key (MSK) of the system. The authority keeps the MSK confidential while publishing PK for use by financial institutions (FIs) and clients. In addition, the authority issues a secret key based on cryptographic methods to each FI, ensuring that the FIs have access to encrypted client data. This entity also acts as an Attribute Authority, validating certain attributes of clients when requested by the FIs.
2. **Clients:** The clients are the users of financial institutions (Bank A or Bank B) who enroll in the blockchain-based e-KYC system. Each client possesses her own cryptographic key pair (Public/Private keys) used to encrypt and decrypt her personal credential data. When clients interact with financial institutions, they must first provide their e-consent, which is digitally signed using their private

key (PrivKeyclient_ID) to allow the FI to store, process, and use their KYC data. This consent is stored on the blockchain for non-repudiation.

3. **Financial Institutions (FIs):** Financial institutions, such as Bank A and Bank B, act as the primary service providers interacting with the clients. They facilitate the e-KYC process by managing client registration, verifying credentials, and ensuring data privacy and security. FIs interact with smart contracts to execute processes like document redaction, data encryption, and KYC verification. They also coordinate with the Attribute Authority to verify the validity of client credentials.
4. **IPFS**: The InterPlanetary File System (IPFS) serves as the distributed storage layer for the encrypted KYC documents. Once the FI receives the client's KYC documents, the files are encrypted and uploaded to the IPFS, generating a hash value (h) using SHA-256. This hash serves as a unique index to retrieve the document from the IPFS storage. The Distributed Hash Table (DHT) stores the mapping between the client's citizen ID and the encrypted document, making it accessible but secure. IPFS ensures that KYC documents are stored in a tamper-resistant manner, enabling easy retrieval when required.
5. **Consortium Blockchain**: In this e-KYC system, a consortium blockchain is utilized to maintain the transaction logs of all KYC-related activities, ensuring integrity, privacy, and transparency across multiple financial institutions (FIs) and stakeholders. The consortium blockchain is jointly governed by a set of trusted financial institutions (such as Bank A and Bank B) and the Attribute Authority, ensuring that no single entity has full control over the network. This ensures a high level of decentralization while maintaining governance by trusted participants.
6. **Smart Contracts(SC):** The system implements five redactable-enabled smart contracts to automate and manage the entire e-KYC lifecycle, from consent generation to document redaction and auditing:

   6.1 Coordinate & Generate e-Consent Smart Contrac(Sc1)t: This smart contract handles the coordination of e-consent generation between the client and the FI. It allows the FI to request the client's consent to process their KYC documents and digitally records the client's signed consent on the blockchain. It also includes a revocation option, allowing clients to withdraw their consent at any point.

   6.2 Register, Encrypt, Upload & Redact KYC Document Smart Contract(Sc2): This smart contract manages client registration, encryption of KYC documents, and their subsequent upload to IPFS. It also handles future redactions of the documents if required. The KYC files are encrypted using an AES session key before being uploaded to IPFS, and the session key is encrypted with the client's public key for added security. The smart contract also stores the encrypted document's hash on the blockchain.

   6.3 KYC Verification Smart Contract(Sc3): The KYC Verification Smart Contract is responsible for verifying the authenticity of the client's KYC documents stored in IPFS. It ensures that the encrypted documents are valid and untampered by comparing the stored hash with the computed hash. This contract also allows verification to proceed even if redaction has occurred, by maintaining redaction awareness.

   6.4 Redaction Management Smart Contract(Sc4): The Redaction Management Smart Contract is used when a client or FI needs to redact specific parts of the stored KYC document. This smart contract facilitates redaction using chameleon hashing, ensuring that the modified data's hash remains consistent with the blockchain records while still reflecting the changes made to the underlying data. After redaction, a proof of redaction is generated and stored on the blockchain to guarantee traceability and accountability.

   6.5 Transparency Auditing Smart Contract(Sc5): This smart contract provides a transparent audit trail of all actions related to the e-KYC process, including data access, consent revocation, document redaction, and verification. Auditors can use this contract to review a comprehensive history of all operations performed on a client's data, ensuring accountability and compliance with regulations such as GDPR.

B. CHAMELEON HASHING

The system utilizes Chameleon Hashing to enable selective modification (redaction) of blockchain data without disrupting its structure or cryptographic integrity, making the blockchain GDPR-compliant, and fulfilling the requirement of the "right to be forgotten."

1. **HGen** $(1^k) \rightarrow (hk, tdk)$ **:** The algorithm generates a trapdoor key (tdk) that enables modifications and a hash key (hk) from a prime number p, its subgroup, and a generator of quadratic residues modulo g.

2. **Hashing:** To hash a message m, the algorithm combines it with a randomly generated value r and applies a collision-resistant hash function H.

$$h = r - (y \cdot H(m||r) + g^{s}) \bmod p$$

This hash value h uniquely identifies m while being tied to randomness r and the public hash key hk.

3. **Collision:** If a message m needs to be updated to m', the trapdoor key (tdk) is used to adjust r so that the new hash value remains unchanged, thus preserving the Blockchain integrity.

$$r' = h + g^{k} \bmod p \text{ where } k \text{ is a new random scalar}$$

$$s' = k - H(m'||r') \cdot x \bmod p \text{ where } x = tdk$$

Obtaining new values r' and s' that the message m' remains the same hash value as the original m.

$$h = Hash(hk, m, r, s) = Hash(hk, m', r', s')$$

B. System Process

Phase 1: Setting up phase

In this phase, the Trusted Authority (TA) initializes the cryptographic variables and deploys the system's smart contracts. Keys are generated using Elliptic Curve Cryptography (ECC) with the P-256 (NIST curve) standard.

1). Master Secret Key (Msk_{α}) The TA generates a random scalar α from the finite field of the curve $Z_{q'}^{*}$ and the public master key is computed by function $Msk_{\alpha} = \alpha$

2). Public Parameters

2.1). Prime order of the elliptic curve group (q)

2.2). Group of elliptic curve points (G)

2.3). Generator point of the elliptic curve (P)

2.4). The TA's public key that corresponded to Master Secret Key ($P_{\alpha} = \alpha P$)

3). Hash Parameters These parameters map inputs to elliptic curve points.

3.1). Hash of metadata of PII on Blockchain (H_{PII}) as H_1

3.2). Key Generation for each financial institute as H_2

3.3). Transformation Key for Redaction as H_3

3.4). Hashed Index of the files stored on IPFS (H_{Index}) as H_4

4). Output

4.1). $params = \{G, q, P, P_{\alpha}, H_1, H_2, H_3, H_4\}$

4.2). Master secret key $Msk_{\alpha} = \alpha$

Phase 2: Key Generation phase

This phase defines the cryptographic keys used by financial institutions (FIs) and data owners. It involves interactions between the Trusted Authority (TA) and FIs to generate secure key pairs and establish relationships.

1). Entity-Side Operations (FI) The Financial institute generates a random value r_{FI_id} for computing pseudo-identifier pid_{FI_id} and compute R_{FI_id} which is then grouped together and sent to the Authority.

1.1). The FI generates a random value $r_{FI\_id} \in Z_{q'}^{*}$

1.2). Compute $R_{FI\_id} = r_{FI\_id} \cdot P$

1.3). Compute the pseudo-identifier $pid_{FI\_id} = H_2((r_{FI\_id} \oplus id_{FI\_id}) \cdot P \,||\, id_{FI\_id}$

1.4). FI sends (R_{FI_id}, pid_{FI_id}) to the Authority

2). Trusted Authority-Side Operations

2.1). Generate a random scalar $r_A \in Z_{q'}^{*}$

2.2). Compute $R_A = r_A \cdot P$

2.3). Derive the certificate $Cert_{FI\_id} = R_{FI\_id} + R_A$

2.4). Compute Auxiliary Scalar $r_{aux} = H_2(Cert_{FI\_id} ||pid_{FI\_id}) \cdot r_A + \alpha$

2.5). Returns $(Cert_{FI\_id}, r_{aux})$ to the FI.

3). Entity-Side Finalization (FI)

3.1). Compute their Private Key. $PrivK_{FI\_id} = H_2(Cert_{FI\_id} ||pid_{FI\_id}) \cdot r_{FI\_id} + r_{aux}$

3.2). Compute their Public Key

$$PubK_{FI\_id} = PrivK_{FI\_id} \cdot P$$

3.3). Verify their public key to make sure it matches the certificate.

$$PubK_{FI\_id} = H_2(Cert_{FI\_id} \,||pid_{FI\_id}) \cdot Cert_{FI\_id} + P_{\alpha}$$

4). Delegate Key Pairs Generation

4.1). Secret key $PrivDK_{FI\_id} \in Z^*_{q'}$

4.2). Public key $PubDK_{FI\_id} = PrivDK_{FI\_id} \cdot P$

5). Symmetric Key Generation

Use the delegate private key and metadata to derive a unique AES symmetric key. This key is stored securely in the FI's IPFS and used for decrypting ciphertext during the e-KYC phase.

$$SymKey = H_2(PrivDK_{FI\_id} \,||\, NatID)$$

Phase 3: Customer Enrollment phase

This phase is where customer information is securely registered in the system and added to the Blockchain. It should be noted that two separate file types were added to the system, and it's the FIs who are the ones adding customer data to the system.

1). Obtaining customer's data Let customers provide their data to register their accounts to the financial institute. In this data, there are 4 variables that they need to provide, grouped together as M.

$$M = \{CustID,\ NatID,\ Name,\ Addr,\ DoB\}$$

2). Customers provide their consent Before adding customer data to the system, the Financial Institute (FI) must obtain explicit consent from the customer to be able to comply with the GDPR law. This consent can include

2.1). Permission to store and process data on the blockchain and IPFS.

2.2). Details on how their data will be used.

2.3). Rights to revoke consent at any time.

In this step, consent is digitally signed by the customer $Consent$.

$$Consent = \{ConsentID,\ CustID,\ NatID,\ CustSig\}$$

Once all the information is provided, the Financial Institute shall process the customer data.

Phase 4: Data uploading phase

Encryption Phase Each FIs maintains its own IPFS node to store customer data. Only a lightweight record is uploaded to the Blockchain, sufficient to identify customers and their relationship with the FI they've registered with. This lightweight record is called ($meta$). The encryption phase uses these variables to encrypt the Metadata ($meta$) of Personally Identifiable Information of **each customer** (M) where

$$M = \{\, Customer\ ID,\ National\ ID,\ Name,\ Address,\ Date\ of\ Birth,$$

1). Personally Identifiable Information (M) **Encryption process.**

1.1). Use a cryptographically secure random number generator to generate a unique AES symmetric key. It's tied to the delegate key and metadata. This key is used for decrypting the cipher text and is sent to another FI in e-KYC phase.

2). Blockchain Transaction Encryption, upload process This algorithm takes the public parameter ($params$), few attributes from M, Secret key of Financial Institute Bank A ($PrivK_{FI\_BankA}$), and Secret delegate key ($PrivDK_{FI\_BankA}$) as an input for encryption, encrypted as metadata. This transaction is included with and is published to the Blockchain.

2.1). Compute r, a random value that is used to generate a cryptographic commitment tied to ($meta$). It's computed using this function.

$$r = H_1(PrivDK_{FI\_BankA} \,||\, M)$$

2.2). Compute a point on an elliptic curve using the dot product function.

$$R = r \cdot P$$

2.3). Once r and R are computed, we use R to derive a shared secret for encrypting the metadata ($meta$)

$$C_{FI\_BankA} = M \oplus H_1(R \,||\, PrivDK_{FI\_BankA})$$

2.4). Compute a hash of C_{FI_BankA} and $meta$ for tamper-proofing.

$$h_{FI\_BankA} = H_1(C_{FI\_BankA} \,||\, M)$$

2.5). Derive signature components as Sig_{FI_BankA} to bind $PrivK_{FI\_BankA}$ and r

$$Sig_{FI\_BankA} = r - h_{FI\_BankA} \cdot PrivK_{FI\_BankA}$$

2.6). Combine all components into a hashed transaction $CustomerTX$ and upload it to Blockchain.

$$CustomerTX = H_1(NatID, Consent, CID, h_{FI\_BankA}, Sig_{FI\_Ba}$$

The variables for identification in e-KYC phase are National ID ($NatID$), Customer consent ($Consent$), Referenced Index file on FI's IPFS (CID), hash function and Signature of the FI who is responsible for the registration process of the customer.

Decryption Phase The algorithm decrypts the encrypted customer data CT_M using the symmetric encryption key $SymKey$ and returns the original message M (e.g.PII) that was encrypted. In this part, we assume that it's Bank A who encrypts and decrypts.

1). Transaction Decryption process Before we can decrypt M**,** we have to retrieve a value like CID which is necessary to find the file location on IPFS**.**

1.1). First could use unique identifier such as $NatID$ to locate the corresponding $CustomerTX$ on the Blockchain. Here we need to extract the necessary field (CID).

$$CustomerTX = \{NatID, Consent, CID, h_{FI\_BankA}, Sig_{FI\_BankA}\}$$

1.2). Verify the Transaction Integrity by recomputing hash for validation.

$$h'_{FI\_BankA} = H_1(C_{FI\_BankA} \,||\, M)$$

Check if

$$h'_{FI\_BankA} = h_{FI\_BankA}$$

If the hashes don't match, terminate the process as the metadata is invalid or has been tampered with.

2). Personally Identifiable Information (M) Retrieval Process

2.1). Use the CID retrieved from $CustomerTX$ to locate and retrieve the encrypted customer information (CT_M) from Bank A's private IPFS node.

$$IPFS(CT_M) \rightarrow CID$$

3). Personally Identifiable Information (M) Decryption Process

3.1). To decrypt the personally Identifiable Information, Bank A can decrypt the CT_M that they stored inside their database.

$$M = DEC_{AES}(SymKey \,||\, CT_M)$$

3.2). However, if Bank A doesn't store their key inside a database (for security reasons), they can regenerate a Symmetric Key with this function again.

$$SymKey' = H_2(PrivDK_{FI\_id} \,||\, NatID)$$

$$M = DEC_{AES}(SymKey' \,||\, CT_M)$$

Phase 5: e-KYC process phase

The system facilitates secure and privacy-preserving data sharing between financial institutions (FIs) using zero-knowledge proofs (ZKP) and encryption. This updated protocol includes FI Bank A (source financial institution) and FI Bank B (destination financial institution) in the design. It supports ZK-Rollup for efficient proof aggregation and sharding for distributed proof verification while enabling secure and scalable data sharing between financial institutions.

1. System Setup

Initialize cryptographic parameters, shard configurations, and inter-FI trust.

1.1). Trusted Setup for ZKP

To enable zero-knowledge proofs (ZKP), zk-SNARKs is used for succinct proofs within a trusted or transparent setup. During this process, proving keys pk and verifying keys vk are generated as:

$$\{pk, vk\} \leftarrow Setup(\lambda) The\ keys\ pk$$

and vk are then published for use by FIs and customers.

1.2). Shard Assignment

Users are divided into multiple shards, represented as:

$$S_1, S_2, \ldots, S_n$$

Each shard is managed by a verifier node (VN_i), and each shard maintains its own local Merkle tree to store user proofs securely.

1.3). Inter-FI Key Exchange
To establish secure data sharing between financial institutions, FI_{BankA} and FI_{BankB} exchange their public keys:

$$PubK_{FI_{BankA}}, PubK_{FI_{BankB}}$$

1.4). Public Parameters
The final step involves publishing public parameters that define the cryptographic and structural setup of the system. These parameters are represented as:

$$params = \{G, q, P, H, pk, vk, n\}$$

2. User Authentication with FI Bank A
This process aims to authenticate the customer with FI A using ZKP and shard-based verification.

2.1). Proof Generation
The customer generates a ZKP to prove ownership of their private key $PrivK_C$ without revealing it. This proof (π_C) is created through the following steps: A random scalar r is selected, where:

$$r \in Z_C^*$$

Next, a commitment is calculated as:

$$R = r \cdot P$$

A challenge is then generated using a hash function, combining the customer's ID, the commitment R, and the system parameters:

$$c = H(CustID||R||params)$$

The response is computed as:

$$s = r + c \cdot PrivK_C \bmod q$$

Finally, the output of this process is the proof:

$$\pi_C = CustID, R, s$$

2.2). Shard-Based Verification
The customer submits the proof (π_C) to their assigned shard verifier (VN_i). The verifier checks the proof using the equation:

$$s \cdot P = R + c \cdot PubK_C$$

After verification, the proof is added to a Merkle tree, and the shard root is updated as:

$$root_i = H(R_1||R_2||...||R_m)$$

2.3). Batch Aggregation
Shard proofs are aggregated into a ZK-Rollup proof (π_{rollup}) and submitted to the blockchain along with the global root:

$$root_{global}, \pi_{rollup}$$

3. Customer Request for Data Sharing
This process allows the customer to request FI Bank A to share their data with FI Bank B.

3.1). Consent Submission
The customer creates a consent message, which includes:

$$Consent = \{ConsentID, CustID, FI_{BankA}, FI_{BankB}, Timestamp, Cust...$$

The consent message is encrypted using FI A's public key:

$$Enc(Consent) = ENC_{PubK_{FI_{BankA}}}(Consent)$$

The customer then sends the encrypted consent message $Enc(Consent)$ to FI Bank B.

3.2). FI Bank B Authentication
The customer authenticates with FI Bank B using zero-knowledge proof (π_C). FI Bank B verifies the proof following the same steps as in the previous phase. Upon successful authentication, FI B forwards the encrypted consent message $Enc(Consent)$ to FI Bank A.

4. FI Bank A Data Preparation
This process is for FI Bank A to prepare and re-encrypt customer data for FI Bank B.

4.1). Consent Validation
FI Bank A decrypts the consent using its private key:

$$Consent = DEC_{PrivK_{FI\_BankA}}(Enc(Consent))$$

FI Bank A validates the signature ($CustSig$) and checks the details, such as ($CustID$, FI_BankB, etc).

4.2). Data Retrieval
FI Bank A retrieves the encrypted customer data from the IPFS or blockchain:

$$\{Enc(M),\ Enc(SymKey)\} = BlockChain.query(CustID)$$

4.3). Data Re-encryption
FI Bank A re-encrypts the symmetric key ($SymKey$) for FI Bank B using FI Bank B's public key:

$$Enc_{Symkey} = ENC_{PubK_{FI\_BankB}}(SymKey)$$

4.4). Transformation Key Generation
FI Bank A generates a transformation key ($tk_{A\to B}$) for metadata:

$$tk_{A\to B} = H_3(meta \,||\, r \cdot PrivDK_{FI\_BankA} \cdot PubK_{FI\_BankA}) \oplus H_3(1$$

4.5). Transformed Ciphertext
FI Bank A transforms the encrypted metadata and outputs:

$$CT'meta = C_{FI\_BankB},\ meta,\ pid_{FI\_BankB},\ h_{FI\_BankA},\ Sig_{FI\_BankA}$$

5. Data Transfer to FI Bank B
This process is for FI Bank A to securely transfer data to FI Bank B.

5.1). Data Transmission
FI Bank A transmits the following items to FI Bank B:

Transformed ciphertext ($CT'meta$).

Re-encrypted symmetric key (Enc_{Symkey}).

Encrypted PII file ($ENC(M)$).

5.2). Proof Submission

FI Bank A submits a zero-knowledge proof of transformation ($\pi_{A\to B}$) to the blockchain.

6). FI Bank B Verification and Decryption

This process is for FI Bank B to verify the data and decrypt it.

6.1). Symmetric Key Decryption
FI Bank B decrypts the symmetric key using its private key:

$$SymKey = DEC_{PrivK_{FI\_BankB}}(Enc_{Symkey})$$

6.2). PII Decryption
FI Bank B uses the decrypted symmetric key ($SymKey$) to decrypt the PII file:

$$M = DEC_{AES}(SymKey, ENC(M))$$

6.3). Metadata Verification

FI Bank B verifies the integrity of the metadata using the transformation key ($tk_{A\to B}$).

The authentication level is if B wants to check the data of a customer that is really from A or not, B has to request for verification, including name or citizenID, and sign with Bank B private key to ensure Bank A that real Bank B has requested to check data. Then A will look into the blockchain to find an index (check from the Hash value of CitizenID, if the blockchain has the hash value then we can say that this data is the customer of Bank A) or Bank B can check by themself on the blockchain.

1). Level 1 verification
1.1). Customers provide their information to Bank B, this information is stored by Bank A on Consortium Blockchain. The data that is stored are

$$CustomerTX = \{NatID,\ Consent,\ CID,\ h_{FI\_BankA},\ Sig_{FI\_BankA}\}$$

Customers can provide their $NatID$ and which back they've registered with. (In this case, Bank A.

1.2). Bank B locates the stored transaction on Blockchain, retrieving variables like $NatID$, $Consent$, Sig_{FI_BankA}. This Sig_{FI_BankA} is used to verify if Bank A is the one who register this customer.

2). Level 2 verification

The second level required the original FI to send customer data to the FI that was requested. This step required encrypting the Symmetric Key with Bank B's public key where the original $SymKey$ is.

$$SymKey = H_2(PrivDK_{FI\_id} \,||\, NatID)$$

2.1). Encrypt $SymKey$ with Bank B's public key

$$EncSymKey = ENC_{AES}(PubDK_{FI\_BankB} \,||\, Symkey)$$

2.2). Pass $EncSymKey$ and CT_M to Bank B.

Phase 6: Optimized Redaction Phase with Optimized Techniques

The redaction phase leverages off-chain processing for scalable and efficient data modification and dynamic key rotation using threshold cryptography to enhance security. The following algorithm details the steps to implement the optimized redaction phase in which the updated Chameleon Hash is computed off-chain. in compliance with privacy regulations such as GDPR. This phase is executed by the Redaction smart contract. There are five steps as follows:

Step 1: Validate Redaction Request: The Redaction smart contract verifies whether the transaction exists in the blockchain. If it exists, the customer's consent and the customer's digital signature are verified through the following functions.

```
Exists(Tx_ID) = True
if Tx_ID in Blockchain
else False
VerifySignature(Consent, CustSig, PubK_Cust) = True
```

Step 2: Retrieve Chameleon Hash:

The smart contract retrieves the original transaction, including the data (m), randomization factor (r), and the Chameleon Hash (H_{CH}). In this step, r is decrypted off-chain using the trapdoor key shares.

- Retrieve transaction components:
  $T_x = \{m, r, H_{CH}(m, r)\}$

- Decrypt the randomization factor:
  $r = Decrypt(Tdk_1, Tdk_2, ..., Tdk_t)$

Step 3: Generate Controlled Collision:

A new randomization factor (r') is generated, and the updated Chameleon Hash is computed off-chain. The hash consistency is verified to ensure integrity.

- Generate a new randomization factor:
  r' = GenerateRandom()

- Compute the updated Chameleon Hash:
  $H_{CH}(m', r') = H(m') + f(r', PubK)$
  Ensure: $H_{CH}(m', r') = H_{CH}(m, r)$

Step 4: Key Rotation (Optional):

In this step, we introduce Key rotation method using Threshold cryptography to ensure long-term security by periodically updating the trapdoor key (Tdk).

Threshold cryptography splits the current trapdoor key into shares ($Tdk_1, Tdk_2, ..., Tdk_n$) distributed among n participants. A threshold t of these shares is required to reconstruct the key for secure rotation. The new key is then generated, split into shares, and securely redistributed based on Shamir's Secret Sharing.

The below pseudocode presents the key rotation procedure:

```
def rotate_trapdoor_key(current_shares, threshold):

   # Inputs:
- current_shares: List of trapdoor key shares { Tdk_1, Tdk_2, ..., Tdk_n }
- threshold: Minimum number of shares required to reconstruct Td_k (denoted as t)
- n: Total number of participants

   # Outputs:
- new_shares: New trapdoor key shares {Tdk_new,1, Tdk_new,2, ..., Tdk_new,n}
- Tdk_old: Archived old trapdoor key
   # Step 1: Agreement Phase
   agreeing_shares =
collect_agreeing_shares(current_shares, threshold)
   if len(agreeing_shares) < threshold:
```

```
    return "Threshold not met for key rotation"

  # Step 2: Reconstruct Current Key
  Tdk = reconstruct_key(agreeing_shares)
   Reconstruct using threshold cryptography:
  Tdk = Combine({Tdk1, Tdk2, ..., Tdkt })

  # Step 3: Generate New Key
  Tdk_new = generate_random_key()

   # Step 4: Create New Shares
   Use Shamir's Secret Sharing to split Tdknew into n shares
  new_shares = []
  coefficients = [Tdknew] + [random.randint(1, prime_field) for _ in range(threshold - 1)]
  for i in range(1, len(participants) + 1):  # Assuming participants are indexed 1 to n
      x = i
      y = sum(coeff * (x ** power) for power, coeff in enumerate(coefficients)) % prime_field
      new_shares.append((x, y))  # Share is (x, y)

  # Step 5: Secure Share Distribution
  for participant, share in zip(participants, new_shares):
      encrypted_share = encrypt_with_public_key(participant.public_key, share)
      send_to_participant(participant, encrypted_share)  # Securely distribute encrypted shares

  # Step 6: Archive Old Key
  archive_key(Tdk)  # Securely archive the old trapdoor key for backward compatibility

  return new_shares, Tdknew
```

original hash. The transaction replacement and redaction proof are done via the following functions:

$$Tx' = \{m', r', H_{CH}(m', r')\}$$
$$Prove(H_{CH}(m, r) = H_{CH}(m', r'))$$

Step 6: Audit and Logging:
An off-chain audit log records the redaction event. A hash of the audit log is stored on-chain for integrity and transparency.

$$AuditLog = \{Tx_{ID}, Consent, 'Redaction', Timestamp, AuthorizedEntities\}$$
$$AuditHash = H(AuditLog)$$

REFERENCES

[1] General Data Protection Regulation (GDPR), https://gdpr-info.eu, Accessed 8 Jan 2025

[2] S. Fugkeaw, "Enabling Trust and Privacy-Preserving e-KYC System Using Blockchain," in *IEEE Access*, vol. 10, pp. 49028-49039, 2022, doi: 10.1109/ACCESS.2022.3172973.

[3] Y. Suga, "ICH 3-party model for claims distribution on the blockchain and their application to the e-KYC-e model," 2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech), Osaka, Japan, 2022, pp. 512-513, doi: 10.1109/LifeTech53646.2022.9754919.

[4] B. Jaber, D. Kriwiesh and M. A. AlRagheb, "A Blockchain Framework in the Banking Sector Based in e-KYC System Conceptual Framework," 2024 2nd International Conference on Cyber Resilience (ICCR), Dubai, United Arab Emirates, 2024, pp. 1-4, doi: 10.1109/ICCR61006.2024.10532852.

[5] Feulner, S., Schlatt, V., Guggenberger, T., Völter, F., Stoetzer, JC. (2024). Self-Sovereign Identity for Digital KYC. In: Fridgen, G., Guggenberger, T., Sedlmeir, J., Urbach, N. (eds) Decentralization Technologies. Financial Innovation and Technology. Springer, Cham. https://doi.org/10.1007/978-3-031-66047-4_7

[6] G. M. Faruk Ahmed, Imran Hasan, Md Soumike Hassan, Rahul Khan, Abu Jor Al Gefari, Zain Buksh, Joseph Liu, A B M Shawkat Ali, "Enhancing e-KYC Security and Privacy: Harnessing Quantum Computing and Blockchain in Web 3.0", Distributed Ledger Technologies: Research and Practice, 2024, doi/10.1145/3686166.

[7] Pradnya Patil, M. Sangeetha, Blockchain-based Decentralized KYC Verification Framework for Banks, Procedia Computer Science,Volume 215, 2022, Pages 529-536.

[8] M. Kumar, P. A. Nikhil, and P. Anand, "A blockchain based approach for an efficient secure kyc process with data sovereignty," Int J Sci Technol Res, vol. 9, pp. 3403–3407, 2020

[9] A. A. Mamun, S. R. Hasan, M. S. Bhuiyan, M. S. Kaiser and M. A. Yousuf, "Secure and Transparent KYC for Banking System Using IPFS and Blockchain Technology," *2020 IEEE Region 10 Symposium (TENSYMP)*, Dhaka, Bangladesh, 2020, pp. 348-351, doi: 10.1109/TENSYMP50017.2020.9230987.

[10] S. Bhatia, L. Vishwakarma and D. Das, "Cost-efficient Blockchain-based e-KYC Platform using Biometric verification," *2024 International Conference on Information Networking (ICOIN)*, Ho Chi Minh City, Vietnam, 2024, pp. 403-408, doi: 10.1109/ICOIN59985.2024.10572111.

[11] Schlatt, V., Sedlmeir, J., Feulner, S., & Urbach, N. (2022). Designing a framework for digital KYC processes built on blockchain-based self-sovereign identity. Information & Management, 59. https://doi.org/10.1016/j.im.2021.103553

[12] Patkar, M., Giri, A., Gite, P., Shinde, A., Shetye, K. (2024). Privacy Preserving and Trustworthy E-KYC System Using Blockchain. In: Mishra, D., Yang, X.S., Unal, A., Jat, D.S. (eds) Data Science and Big Data Analytics. IDBA 2023. Data-Intensive Research. Springer, Singapore. https://doi.org/10.1007/978-981-99-9179-2_49

[13] K. Takaragi, T. Kubota, S. Wohlgemuth, K. Umezawa, and H. Koyanagi. 2023. Secure Revocation Features in eKYC-Privacy Protection in Central Bank Digital Currency. In *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences*, 106, 3 (2023), 325–332.

[14] L. -Y. Yeh, W. -H. Hsu and C. -Y. Shen, "GDPR-Compliant Personal Health Record Sharing Mechanism With Redactable Blockchain and Revocable IPFS," in *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 3342-3356, July-Aug. 2024, doi: 10.1109/TDSC.2023.3325907.

[15] G. Zhou, X. Ding, H. Han and A. Zhu, "Fine-Grained Redactable Blockchain Using Trapdoor-Hash," in *IEEE Internet of Things Journal*, vol. 10, no. 24, pp. 21203-21216, 15 Dec.15, 2023, doi: 10.1109/JIOT.2023.3279434.

[16] J. W. Heo, G. Ramachandran and R. Jurdak, "Decentralised Redactable Blockchain: A Privacy-Preserving Approach to Addressing Identity Tracing Challenges," *2024 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, Dublin, Ireland, 2024, pp. 215-219, doi: 10.1109/ICBC59979.2024.10634438.

[17] H. Guo, W. Li, C. Meese and M. Nejad, "Decentralized Electronic Health Records Management via Redactable Blockchain and Revocable IPFS," *2024 IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, Wilmington, DE, USA, 2024, pp. 167-171, doi: 10.1109/CHASE60773.2024.00028.

[18] Yueyan Dong, Yifang Li, Ye Cheng, Dongxiao Yu, Redactable consortium blockchain with access control: Leveraging chameleon hash and multi-authority attribute-based encryption, High-Confidence Computing,Volume 4, Issue 1, 2024,100168

[19] Y. Zhang, Z. Ma, S. Luo and P. Duan, "Dynamic Trust-Based Redactable Blockchain Supporting Update and Traceability," in *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 821-834, 2024, doi: 10.1109/TIFS.2023.3326379.

[20] L. -E. Wang, Y. Wei, P. Liu and X. Li, "Secure and Trusted Copyright Protection for Educational Data on Redactable Blockchains," *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*, Ocean Flower Island, China, 2023, pp. 617-622, doi: 10.1109/ICPADS60453.2023.00096.

[21] S. Xu, J. Ning, X. Li, J. Yuan, X. Huang and R. H. Deng, "A Privacy-Preserving and Redactable Healthcare Blockchain System," in *IEEE Transactions on Services Computing*, vol. 17, no. 2, pp. 364-377, March-April 2024, doi: 10.1109/TSC.2024.3356595