https://doi.org/10.31891/csit-2025-4-2
UDC 004.9

KYSIL Volodymyr, KYSIL Tetiana
Khmelnytskyi National University

APPROACH TO A DECENTRALIZED, PHYSICIAN-ORIENTED EHR ARCHITECTURE WITH CRYPTOGRAPHIC PROTECTION

In modern Electronic Health Record (EHR) systems, electronic records play a primary role in storing examination data; however, classic centralized storage systems face challenges regarding security, privacy, and data availability. Centralized databases are vulnerable to cyberattacks, data leaks, and interoperability issues with other systems, threatening patient confidentiality as well as the efficiency and transparency of medical institutions. In the specific case of a distributed network architecture where servers may operate autonomously without constant internet connectivity, decentralized solutions are required that combine cryptographic protection with ease of use for medical personnel. Physician-centric systems allow for workflow optimization in institutions where doctors collaborate within a trusted environment, while adhering to ethical standards of transparency for patients.

The objective is to create a flexible, autonomous platform that ensures cryptographic data protection through envelope encryption with combined Data Encryption Keys (DEK), server key rotation, and hash chains for modification detection. The system supports profile migration between nodes, the exchange of signed data between physicians, and resource optimization by offloading completed records to a registry node. The primary focus is on patient transparency: any data decryption must be accompanied by a notification to the patient via an active notification system; if the notification system is unavailable, the decryption operation is not performed. Consequently, data decryption is logged (identifying the entity performing the decryption) for future reference.

The system workflow begins with the creation of a patient profile on the physician's local node, where data is encrypted using a combined DEK (static user key + daily node key). Hash chains ensure file integrity, preventing undetected changes, while two differently encrypted copies of the key allow for access recovery by an administrator. This makes the system resilient to failures, with minimal requirements for users, maintaining a balance between efficiency for medical personnel and patient rights regarding access information, data access, and data portability.

Performance evaluation of the proposed architecture demonstrates that the storage overhead for cryptographic metadata is approximately 248 bytes per user profile (server-encrypted key, user-encrypted key, key-encrypted identifier, creation time; the size of encrypted data depends on the algorithm) and 40 bytes (SHA-256 standard + timestamp) per record to ensure cryptographic linkage. This is a negligible amount (<0.001%) compared to the volume of medical data (up to 32 MB per record with images). The absence of a need for global consensus (unlike blockchain solutions) ensures simple O(1) write operations, guaranteeing high performance even on resource-constrained hardware. Efficiency assessment indicates that the architecture adds minimal overhead: only a slight increase in size and access time within the constraints. Thanks to the local consensus model, the system does not require network synchronization, ensuring O(1) write complexity and high speed even on personal devices without dedicated server hardware.

Keywords: EHR, decentralized system, cryptographic protection, hash chains, patient data transparency.

КИСІЛЬ Володимир, КИСІЛЬ Тетяна
Хмельницький національний університет

ПІДХІД ДО ДЕЦЕНТРАЛІЗОВАНОЇ ЛІКАР-ОРІЄНТОВАНОЇ АРХІТЕКТУРИ EHR ІЗ КРИПТОГРАФІЧНИМ ЗАХИСТОМ

В сучасних системах для збереження медичних записів (Electronic Health Records, EHR) електронні записи відіграють основну роль для збереження даних обстежень, проте класичні централізовані системи збереження медичних записів стикаються з проблемами забезпеченням безпеки, приватності та доступності даних. Централізовані бази даних вразливі до кібератак, витоків інформації та проблем з обміном інформацією з іншими системами, що загрожує конфіденційності пацієнтів і ефективності та прозорості роботи медичних установ. У даному конкретному випадку архітектури розподіленої мережі, де сервери можуть працювати автономно без постійного інтернет-з'єднання, потрібні децентралізовані рішення, що поєднують криптографічний захист з простотою використання для медичного персоналу. Лікар-орієнтовані системи, орієнтовані на постачальників послуг, дозволяють оптимізувати процеси в установах, де лікарі співпрацюють у довіреному середовищі, але з урахуванням етичних норм прозорості для пацієнтів.

Метою є створення гнучкої, автономної платформи, що забезпечує криптографічний захист даних через envelope encryption з комбінованими ключами (Data Encryption Key, DEK), ротацію серверних ключів і хеш-ланцюги для виявлення модифікацій. Система підтримує міграцію профілів між вузлами, обмін підписаними даними між лікарями та оптимізацію ресурсів шляхом скидання неактуальних записів у ноду реєстратури. Основний фокус тут робиться на прозорість для пацієнта: при розшифруванні даних ким завгодно пацієнт повинен бути сповіщений про доступ до даних через діючу систему сповіщення - якщо система сповіщення недоступна то розшифровки не відбудеться. Відповідно розшифровка даних для користувача логується (ким було розшифровано) для подальшої інформативності.

Процес роботи системи починається зі створення профілю пацієнта на локальній ноді лікаря, де дані шифруються комбінованим DEK (статичний користувацький ключ + добовий ключ ноди). Хеш-ланцюги забезпечують цілісність файлів, унеможливлюючи непомітні зміни, а дві по-різному шифровані копії ключа дозволяють відновлення доступу адміністратором. Це робить систему стійкою до збоїв, з мінімальними вимогами до користувачів, але з балансом між ефективністю для медичного персоналу та правами пацієнтів на інформацію про доступ та доступ до власних даних і можливістю міграції даних..

Оцінка ефективності запропонованої архітектури показує, що накладні витрати на зберігання криптографічних метаданих становлять приблизно 248 байт на профіль користувача (ключ зашифрований сервером, ключ шифрований користувачем, ідентифікатор шифрованийи кючем, час створення, розмір шифрованих даних залежить від алгоритму) та 40 *байтів (стандарт SHA-256 + час запису) на запис для забезпечення криптографічної зв'язності, що є нехтувано малою величиною (<0,001%) порівняно з обсягом медичних даних (до 32 мб на запис з зображеннями). Відсутність необхідності досягнення глобального консенсусу (на відміну від блокчейн-рішень) забезпечує простоту операцій запису O(1), гарантуючи високу швидкодію навіть на обладнанні з обмеженими ресурсами. Оцінка ефективності показує, що архітектура додає мінімальні накладні витрати: лише збільшення розміру та незначне збільшення часу доступу згідно з врахуванням обмежень. Завдяки моделі локального консенсусу, система не потребує мережевої синхронізації, забезпечуючи складність запису O(1) та високу швидкодію навіть на персональних пристроях без виділеного серверного обладнання.*

Ключові слова: EHR, децентралізована система, криптографічний захист, хеш-ланцюги, прозорість поводження з даними пацієнтів.

Received / Стаття надійшла до редакції 21.09.2025
Accepted / Прийнята до друку 29.11.2025

Introduction

Storage of medical data in distributed information systems requires combining three fundamental properties: confidentiality, integrity, and availability. In cases where the system is deployed not in a centralized data center but across a series of local servers or workstations—sometimes even on individual doctors' workstations—traditional data protection and key management models become overly complex, insufficiently informative, or too difficult for users to comprehend or utilize. In such environments, servers often operate in partial isolation and may lack constant internet access, while an “administrator” is physically present at each separate node. This simultaneously creates an opportunity for immediate oversight and a challenge regarding protection against unauthorized data access by the administrator themselves. Infrastructure limitations impose requirements for simplicity, autonomy, fault tolerance, as well as clarity and usability, since the user or administrator does not need to know much about the system’s internal operation—it suffices for them to know that the system is secure.

The aim of this work is to construct a data protection architecture based on a combination of a static user fingerprint key, a daily key of a separate server node, cryptographic binding of files and keys into a “hash chain,” and the local use of an administrator master key. Such an architecture presents a viable practical solution to the problems of secure storage, transfer of medical records between servers, and logging/notifying users about access to their data, ensuring the capability to export and import full user profiles without loss of integrity.

The article provides a formal description of the architecture, a justification of cryptographic decisions, and an analysis of the approach’s compliance with modern key management and data protection practices in medical information systems.

The main feature of the approach is the use of a combined Data Encryption Key (DEK), which consists of a user component (USER_DEK) and a regularly changed (typically daily) server key (SERVER_DEK_STRING), ensuring a division of responsibility between the user and the server. Additionally, several encrypted copies of the static key (with different encryption sources) are stored in the user profile, allowing for access recovery by the administrator in case of password loss or when access is needed, while simultaneously informing the user.

The scientific novelty of the obtained results lies in solving the problem of ensuring data confidentiality and integrity under conditions of limited trust and resource isolation. Specifically, a decentralized hybrid cryptographic protection scheme has been developed for the first time, which, unlike traditional approaches, is based on the principle of distributed secret possession between the user and the server. This allows for the elimination of dependence on a centralized Key Management Service (KMS) and the removal of a single point of failure in isolated network segments.

At the same time, the method of data immutability verification has been further developed: the use of a local hash chain mechanism is proposed. This ensures the tamper-evidence property with linear complexity, avoiding the significant overhead costs associated with achieving global consensus inherent in distributed ledger-based systems.

Related works

In modern systems, a number of established methods are used to protect data and resources and to ensure storage integrity. Closest to the proposed approach are envelope encryption models, Key Management Systems (KMS), Hardware Security Modules (HSM), and approaches based on cryptographically linked data storage structures (tamper-evident logs), sometimes implemented via blockchain technologies.

Envelope encryption (AES KEY WRAP) is the most common scheme for storing encrypted data in modern cloud platforms: AWS, Google Cloud, Azure, etc. [1]. The scheme involves using two types of keys: the Data Encryption Key (DEK), which directly encrypts data, and the Key Encryption Key (KEK), which encrypts the DEK. The KEK is usually managed by a specialized system (KMS) and can be rotationally updated, while the DEK is often tied to a specific file or dataset. This approach significantly facilitates key rotation and reduces compromise risks, since a leak of the encrypted DEK does not constitute a data leak in itself, as the key itself cannot be accessed without the master key. In some cases, it is possible to create a data decryption system that operates on the client side [1]. KMS (Key Management System) systems [1] provide centralized key management and often use Hardware Security Modules (HSM) to protect the KEK from unauthorized access. However, they require a constant internet connection or access to centralized infrastructure, which is not always possible in local medical environments [1].

Studies [2, 3, 6] examining blockchain and hash chains for EHR protection demonstrate the potential of cryptographic chained storage for detecting modifications to medical records. Although full-fledged blockchains are often excessive for local systems, the principle of file chaining itself is applied in various simplified forms to reduce computational needs and storage requirements for data and keys. Specifically, the blockchain system is used to identify changes so that, if necessary, one can determine whether a record has been edited relative to the original.

Study [5] describes encrypted information exchange based on a custom-developed system with its own encryption and data transmission architecture. Such an approach is not satisfactory for the architecture being developed, where the use of existing, well-known secure information exchange technologies, such as the HTTPS protocol, is quite sufficient, allowing a user to securely transmit information to another user without additional complexities.

Table 1.

Comparison of architectures/approaches

Characteristics	Cloud EHR (AWS/Azure)	Blockchain EHR (Hyperledger)	Current Approach
Internet access Requriement	Critical (High)	High (Synchronization)	Mitigated (Offline-first)
Computing requirement	Low (Server-side)	HIgh (Mining/Consensus)	Low (Hashing)
Key management	Centralized (KMS)	User wallet	Distributed (Envelope Enc.)
Key compromise resilience	Low (Single volatile point)	Medium	High (Daily key rotation)
Складність запису	Depends on network	O(N) | O(log N)	O(1) (Local)

Unlike blockchain systems that require ledger replication, the proposed architecture minimizes infrastructure dependency by relying on the cryptographic verifiability of local data.

System Requirements

From the perspective of UX and security policies, a significant number of works highlight the conflict between client-side encryption (high privacy but high computer literacy requirements) and server-side encryption (simplicity but requiring better operational control) [5]. In medical systems, the latter option is more frequently chosen due to minimal user action requirements; additionally, operations in this system must be independent of user capabilities. Consequently, this approach was selected for the proposed system [7]. In this specific case, the requirements include high privacy and simplicity for the user, alongside the need for enhanced control.

Overall, existing works [3, 7, 10] confirm the importance of a set of critical specifications, such as: separation of responsibility for keys; storage of keys in encrypted form; the necessity for logging and detection of external modifications; mechanisms for data access recovery in the event of compromise or damage to the key chain, where recovery is independent of the end user; data portability mechanisms available both to the end user and within the local network for trusted servers; encryption mechanisms for sensitive/personal data to ensure protection and prevent decryption in the event of key leakage; and automated creation of user identity data requiring only personal data input for initial identity creation (this data is sensitive and encrypted to remain inaccessible during data leaks).

Based on the above, the system under consideration must function under a set of non-trivial requirements. This system for storing and exchanging user data must be an autonomous, distributed system that ensures confidentiality, integrity, and controlled access in environments with limited network resources and minimal hardware capabilities. It must provide data protection based on cryptographic mechanisms (including envelope encryption), support profile migration between servers, and enable access recovery by an administrator while adhering to transparency and local control requirements.

Formulating requirements for architecture approach

To achieve the stated goals of autonomy and security, the architectural requirements for the system were grouped into three areas:

The main functional and non-functional requirements for the system, categorized by autonomy, security, and architectural goals, are formulated below:

Autonomy and Distributed Nature. The system is designed according to the Offline-First principle. A critical capability is for nodes to function without access to a global network or central repositories. Unlike classic client-server solutions, all operations (authentication, encryption, key generation) are performed on local hardware without the use of external Identity Providers (IdP) or cloud-based KMS. Profile migration between isolated servers is implemented via a unified encrypted container, ensuring interoperability in a heterogeneous environment.

Security and Integrity Model. Data protection is based on the Envelope Encryption scheme, where master keys are stored and rotated exclusively locally. Record integrity is guaranteed by a mechanism of unbroken hash chains, which allows for the automatic detection of unauthorized file modifications (tamper-evidence) during import or opening. The administration model limits privileged access to the physical perimeter of the server or a trusted local network, completely excluding attack vectors via remote management (Remote Root Access).

Operational Reliability and Recovery. Considering the specific usage by personnel with basic training, interaction with cryptographic functions (export/import, decryption) is automated and hidden behind a "single-action" interface. The system provides an Emergency Access Recovery mechanism involving an administrator, with mandatory notification of the data owner, as well as support for key backup to protect against hardware failures.

Table 2.

Current architecture description

Designation	**Symbol**	**Description and Purpose**	**Storage Location**	**Lifecycle**
Local Master Key	K_{master}	Administrator's master key. Used for encrypting daily server keys and user key backups.	Locally with the administrator (not stored in DB). Loaded into RAM during startup/recovery.	Static. Changed only upon administrator compromise.
Daily Server Key	K^{d}_{server}	Daily server key. Ensures temporal binding of data and protection against past attacks (Forward Secrecy).	Stored in encrypted form within the Hash Chain (DB).	Rotated automatically every 24 hours (UTC 00:00). Time is configurable here.
User DEK	K_{user}	Static user data encryption key. Unique for each profile. Encrypts all user files and settings.	Stored in the user profile in encrypted form (Envelope).	Immutable during profile existence. Exists in decrypted form during the session.
Combined DEK	$K_{effective}$	Effective key for AES-256. Result of the cryptographic combination of Kuser and Kserver(d).	Exists exclusively in RAM during read/write operations.	Temporary (destroyed after operation completion).

Further in the text, we will adhere to the terminology presented in Table 2 to avoid ambiguity regarding key roles.

The proposed system implements a local hash-chain mechanism that functions as an internal, cryptographically secure log of server key states. Each element of this chain is formed once daily and contains three main components: a UTC timestamp, a daily server master key (randomly generated), and the hash of the previous chain element. The daily master key is generated using a cryptographically secure pseudo-random number generator (CSPRNG) and subsequently encrypted with the administrator's key. For the first element, a zero previous hash is used; for each subsequent one, the previous hash is calculated as the SHA-256 of the concatenation of the previous block's content and the new daily key [Fig 1]. Keys are also backed up to allow for recovery in the event of a failure. This ensures the immutability and stability of the structure: modifying any historical element renders all subsequent elements invalid; however, it remains possible to utilize matching subsequent hashes to identify the specific block that was altered. This approach also minimizes potential damage when the key sequence is edited.

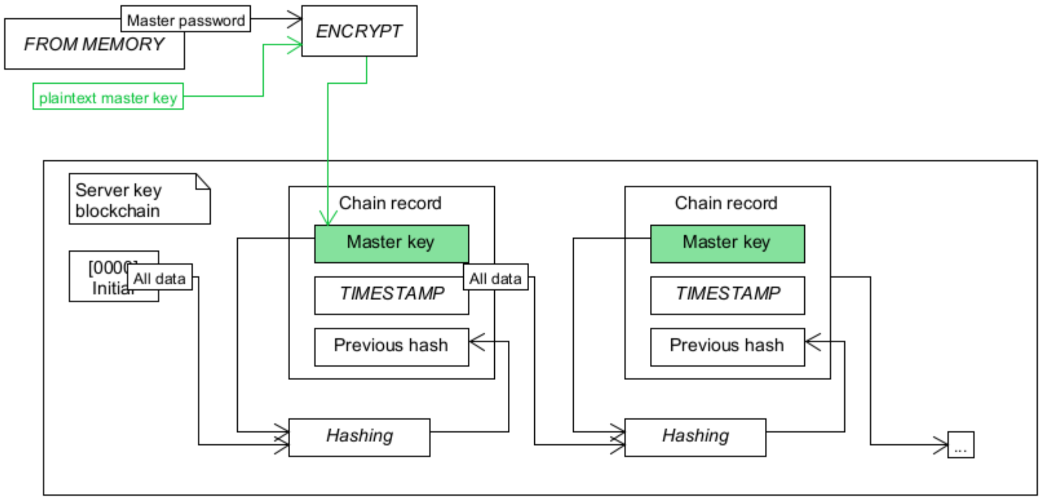

Fig.1. Model of local hash key chain

In the proposed system for secure storage and processing of medical data, a primary role is played by a special key construction scheme based on the combined $K_{effective}$ which is formed as a concatenation of two components: K^{d}_{server}, generated by the server daily and responsible for encrypting data created on that date, and the static K_{user}, unique to each user and immutable throughout the entire lifecycle of their profile [Fig 2]. K_{user} is never stored in plaintext; its availability is ensured by two encrypted copies: a copy encrypted with the user's password for authentication, and a copy encrypted with the daily server key at the moment of profile creation (which in turn is encrypted by the administrator's master key). This allows for access recovery only under the condition of the administrator's physical presence and ensures redundancy without significantly expanding the attack surface. The profile also stores a user identifier encrypted by K_{user}, which is used for authorization based on the profile file without the need to enter inconveniently large hashes.

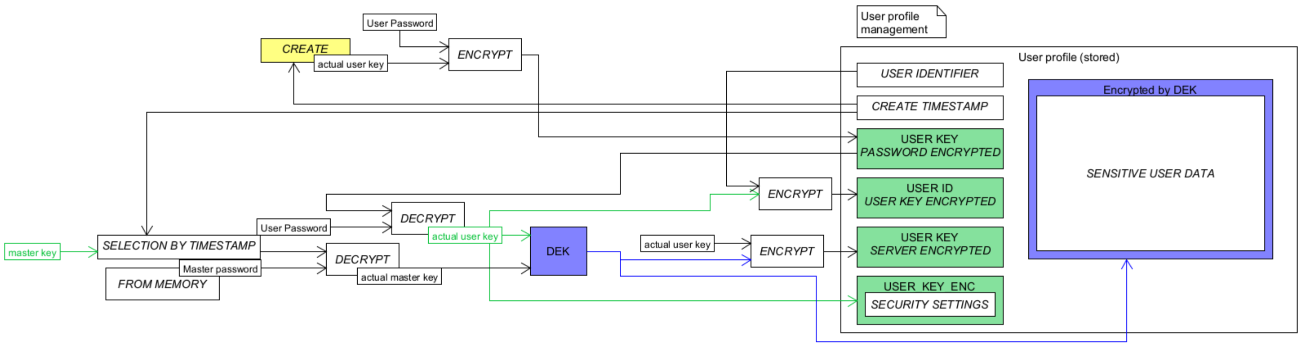

Fig.2. User key system model and DEK for each user and user profile

The main idea is that neither the user nor the administrator ever receives $K_{effective}$ in a decrypted state 1; it exists in plaintext only in RAM during the execution of cryptographic operations2. All profile data and files belonging to the user are encrypted with the corresponding $K_{effective}$ based on their creation date3. To ensure integrity control, a hash chain is used: each file contains a hash that includes the hash of the previous element, and the calculation is performed as $Hash(prev[prev\_hash] + current\_hash)$. This mechanism, analogous to blockchain structures, not only precludes the undetectable modification of individual records but also allows for the unambiguous identification of the point where the sequence was broken, since each block depends on both the previous and the actual data5.

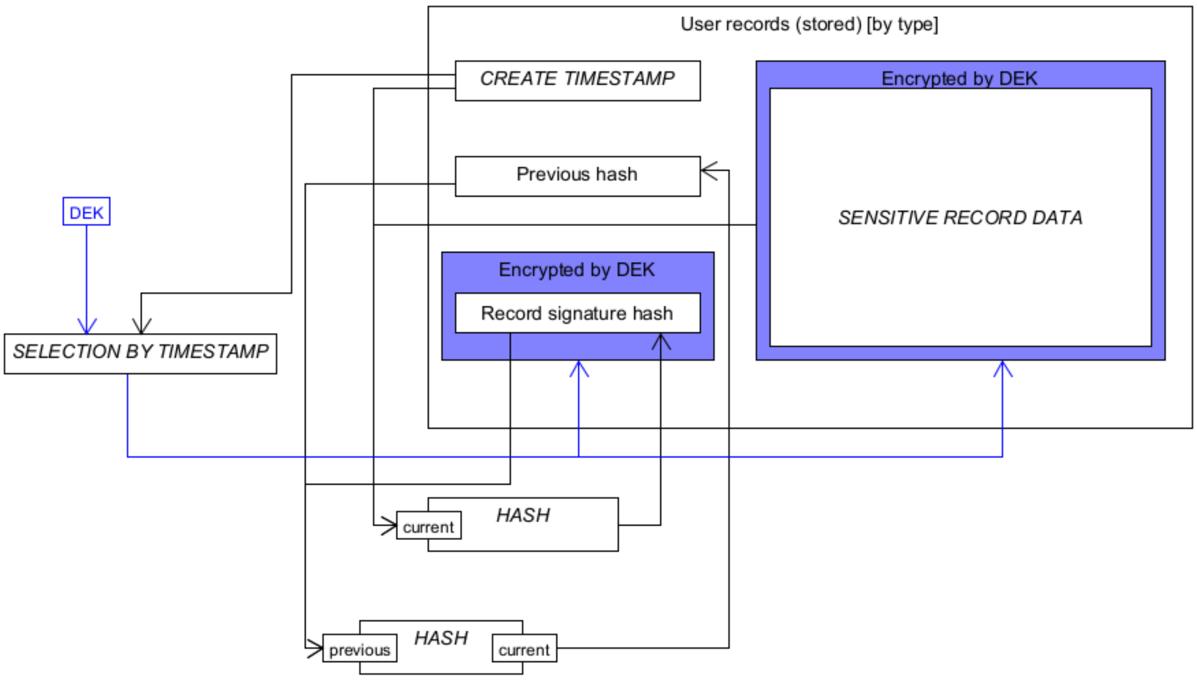

Fig.3. A Model of a System of Verifying Hashes for Medical Records

The user profile also contains encrypted security settings, encrypted by the same K_{user}; these are verified during every decryption operation and cannot be modified without knowledge of the user key, which is never available in plaintext. This eliminates the possibility of altering security policies to bypass protection mechanisms. No mechanism for covert decryption without user notification will be implemented.The data export mechanism is specifically designed considering the low computer literacy of end users and the medical nature of the system. The user is provided with an archive containing fully decrypted data (except for hashes—which are encrypted with the user key, which is in turn encrypted with the user password during export) and an HTML file that serves as a medical chart and integrates local files into a unified viewing interface [Fig 4, 5]. The only element that remains encrypted is the copy of K_{user}, protected by the user password—this ensures authentication during data re-import and preserves security settings [Fig 4, 5]. The procedure involves decrypting K_{user}, fully decrypting the data, generating a directory with the HTML file, re-encrypting K_{user} with the user password, and encrypting the entire archive with the password. Upon import, sensitive data is re-encrypted with the server key.

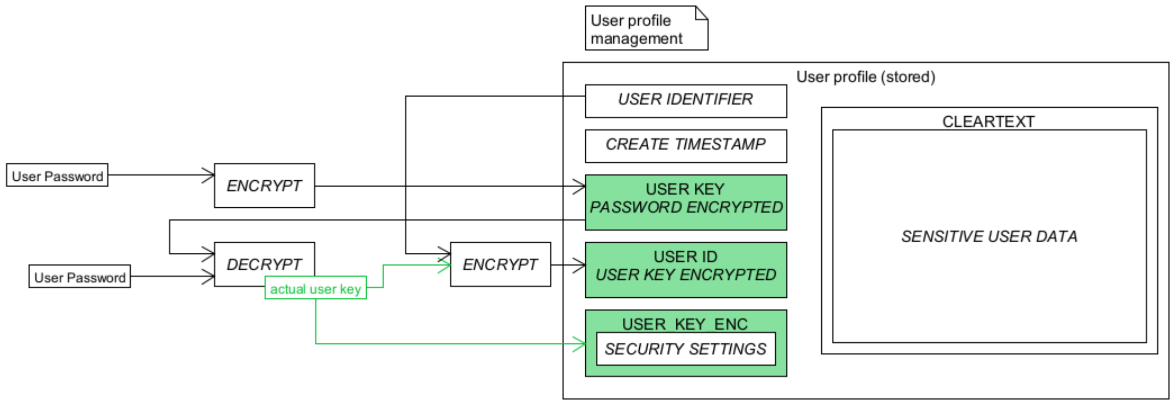

Fig.4. Model of the exported user profile

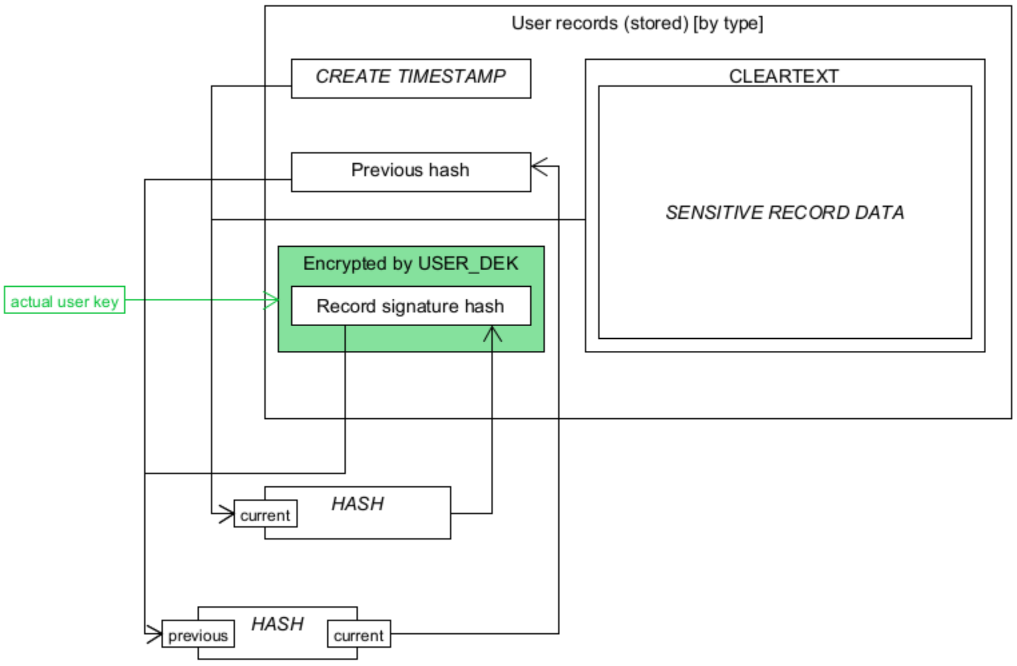

Fig.5. Model of exported user data

Password loss is resolved through an administrator who is physically present at the facility where the server is located. There are two scenarios: either the administrator decrypts K_{user}using K_{master}, fully decrypts the data, and transfers it to the user for import into a new account; or they decrypt K_{user} and simply re-encrypt it with the user's new password. Both procedures exclude the possibility of remote administrator access to the data.

In the threat model, physical access control is key. Since servers are located directly in medical institutions and operate autonomously, attacks require access to the hardware station. The primary risk is the compromise of K_{master} in the event of a file system copy, but this risk is mitigated by the local access model. Compromise of the user password alone does not allow for the recovery of K_{user}, since either K_{master} or the corresponding daily K^{d}_{server} is required. Compromise of K^{d}_{server} limits the consequences to a specific date. MITM attacks are reduced thanks to HTTPS and ring-based administrative access restrictions. Unauthorized file modification is instantly detected through hash chain disruption. Password loss does not lead to data loss thanks to emergency recovery mechanisms.

Architecturally, the system approaches the well-known KEK/DEK model typical of modern KMS solutions: K_{master} acts as the KEK, and $K_{effective}$ is formed by combining K_{user} and K^{d}_{server}. At the same time, the absence of a centralized KMS is a conscious design choice driven by server autonomy. This model allows for the rotation of daily key components without user interaction but complicates the rotation of K_{master}, which is stored locally. In the future, it may be possible to gradually introduce a KEK layer over the existing K_{master} and even migrate K_{master} to a TPM or a local hardware security module if available. Migration to a partial KMS is also possible if a stable network channel becomes available.

From a practical standpoint, the system relies on a combination of technical and organizational measures: offline key backups on external media, administrator access limited exclusively to physical presence and local networks, integrity control via hash chains, limited export policies, and regular rotation of K^d_{server}. Special attention is paid to ethical and legal aspects related to medical data. Since a portion of users do not possess technical skills, the system cannot require manual decryption or complex interactions; therefore, the export mechanism is oriented towards providing data in a fully readable form. The issue of access recovery by an administrator is resolved through the requirement of physical presence, which significantly reduces the risks of abuse and complies with accepted ethical standards for processing medical information.

Security Analysis and Deployment Model

The physician-user acts as the administrator. Security is ensured through physical control of the device. This is the most secure option, as keys do not leave the physical perimeter of the device. For dedicated servers, the system supports secure remote administration exclusively from Trusted Devices. Authentication is performed via an IP whitelist and cryptographic tokens, which resolves the issue of operational access. The only unmitigated vulnerability is the full physical compromise of the machine. Protection is based on the isolation of responsibility: the compromise of one node does not grant access to other physicians' data or the central registry.The architecture utilizes the "WORM" principle for hash chains. Since every record is "sealed" with a daily key, ransomware may block access to files (DoS) but cannot violate their confidentiality or undetectably alter medical history ("retroactively"). The system also includes built-in support for file-based backup of the server key in case hash keys are lost or if there is a requirement for centralized storage of hash keys.

Given the requirement for full node autonomy, the system forgoes automatic server synchronization in favor of a "User-Mediated" data transfer model. The migration protocol consists of three stages: Export, Transport, Import, and Validation. During Export, a Portable Container is formed on the source node. Sensitive metadata (K_{user}) is encrypted with the user's current password. Clinical data is exported in decrypted form (or in a standard exchange format), preserving the hash chain structure for future verification. File transfer is performed physically by the user (USB drive, secure messenger). At this moment, the data is protected by the container, and responsibility for the transmission channel lies with the user. During Import and Validation, integrity verification and owner authentication occur: the receiving node reads the hash chain from the container and recalculates the hashes of all records (H_{calc} = SHA256(...)). If the calculated hash does not match the recorded one, the import is blocked. This guarantees that the user has not modified the data "manually" during transfer; to complete the import, the user must enter their old password (to decrypt K_{user} from the container). After successful validation, the system suggests setting a new local password for this node, with which K_{user} is re-encrypted.

In the case of data transfer to a local concentrator node, decryption occurs at the sender and re-encryption at the receiver, with the exception that new data exists in decrypted form only during direct transfer and is erased afterward, leaving only the encrypted version. This model eliminates the need for mutual server authentication, since trust is built upon the cryptographic container and the user's knowledge of the password, rather than the communication channel.

The system implements a User-Mediated Data Portability model, where the carrier of trust is the cryptographic container, not the network channel. The procedure for import or remote login is based on a cryptographic "handshake" with the container. The transport container holds encrypted user data and a separate security block containing the encrypted control hash of the profile and validation markers. This block is encrypted with the transport password. Upon password entry, the system decrypts the security block, obtaining the control hash. In parallel, hashing of the actual container data is performed. Access is granted exclusively if two criteria are met: the decrypted control hash is identical to the calculated data hash (confirming integrity and password correctness), and the successful decryption of service markers (profile identifier, timestamps), which confirms the validity of the container structure. This approach makes brute-force attacks impossible (since an incorrect password will result in a hash mismatch or invalid markers) and allows the server to authenticate the user without storing their credentials in a local database.

The access recovery procedure (Fig. 6) is designed to ensure Blind Recovery. Since the system uses a file identifier as a key carrier, the administrator acts as a "cryptographic oracle": they provide their master key to temporarily unlock K_{user} in the server's RAM. After the user enters a new password, K_{user} is immediately re-encrypted. Thus, the administrator technically facilitates recovery but does not gain access to the decrypted profile content.

Compliance with ethical standards for patient notification in the system is ensured not by administrative regulations, but at the level of the kernel's software logic (Hard-coded constraint). A "Notification-Gated Decryption" mechanism has been implemented, which makes covert access by the administrator impossible even if they possess all necessary keys.

The access algorithm operates on the principle of an atomic transaction. Upon an access request, the system decrypts a specific portion of the profile in an isolated section of RAM exclusively to read privacy settings ($SecurityPreferences$). If the audit requirement ($NotifyOnDataAccess$) is activated in the user settings, the system performs a health check of the notification subsystem (Email Service Health Check). If the notification service is unavailable (lack of internet connection, SMTP error), the system returns undecrypted data and does not proceed with decryption at all.

Thus, it is technically impossible to obtain decrypted data in "clear" form without successfully initializing the process of informing the data owner. This mitigates the risk of the administrator using technical failures or network disconnections for unauthorized viewing of medical records. However, it is possible to cause a failure in the delivery server if it is local, but this issue already lies within the administrative domain.

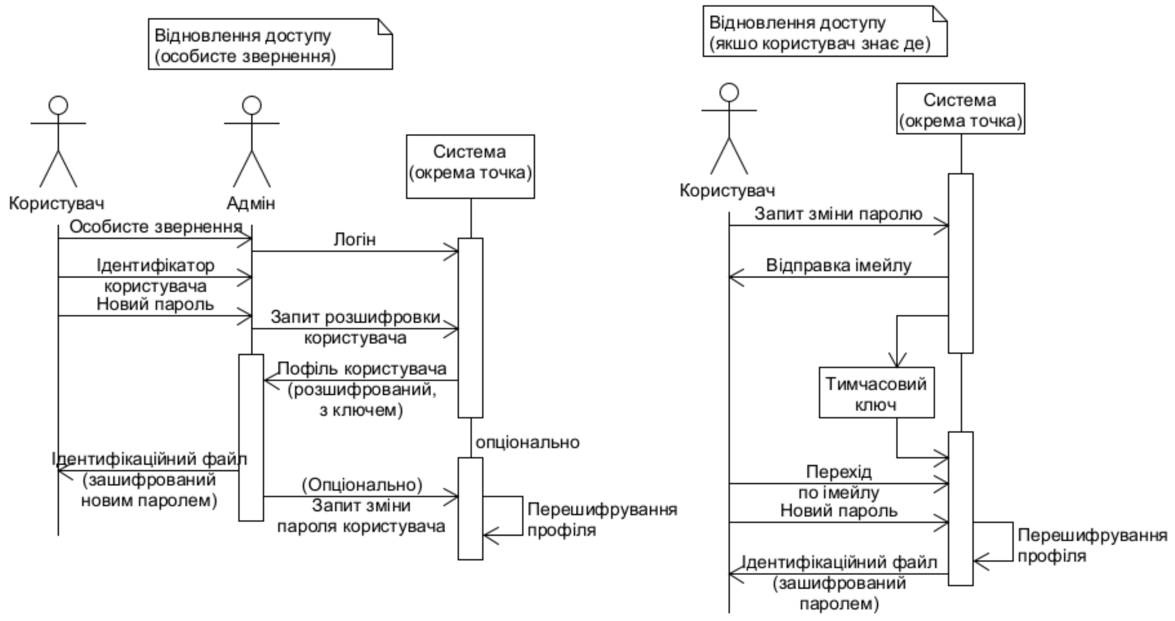
Відновлення доступу (особисте звернення)
Користувач
Адмін
Система (окрема точка)
Особисте звернення
Логін
Ідентифікатор користувача
Новий пароль
Запит розшифровки користувача
Пофіль користувача (розшифрований, з ключем)
опціонально
Ідентифікаційний файл (зашифрований новим паролем)
(Опціонально) Запит зміни пароля користувача
Перешифрування профіля
Відновлення доступу (якшо користувач знає де)
Користувач
Система (окрема точка)
Запит зміни паролю
Відправка імейлу
Тимчасовий ключ
Перехід по імейлу
Новий пароль
Перешифрування профіля
Ідентифікаційний файл (зашифрований паролем)

Fig.6. Flow of access recovery

Implementation and Performance Evaluation

The prototype is implemented on ASP.NET Core (C#) using the System.Security.Cryptography library. The use of a standard stack allows for deployment on any hardware running Windows/Linux without the need for specialized Hardware Security Modules (HSM).

To ensure integrity and confidentiality, the following industry standards were selected:

- Encryption: AES-256 (key 256 bits).
- Hashing: SHA-256 (block size 32 bytes).

The total overhead related to the increase in encrypted content size is excluded from the analysis due to the universality of the algorithm: when comparing different approaches utilizing identical algorithms, the size growth at key junctures remains constant; thus, the difference is confined solely to the additional memory allocated for encryption keys.

Data structure analysis indicates that the cumulative storage overhead for metadata (hashed keys, IV, SHA-256 hash) amounts to approximately 240–250 bytes per user profile. Additionally, 32 bytes are allocated specifically for the hash maintaining the user's data sequence integrity chain. In comparison with typical medical data (images, scans > 20 MB), this overhead (<0.001%) is statistically insignificant, confirming the efficiency of the selected approach for local systems with limited storage.

Discussion

The use of Distributed Ledger Technology (DLT) for isolated medical offices creates excessive network and storage overhead. The proposed approach replaces "global consensus" with local cryptographic verification (via SHA-256 chains). This enables privacy (data does not leave the system without permission) and O(1) write performance, which is unachievable for public blockchains due to the necessity of node synchronization. Local hash chain integrity verification is available upon request.

A similar approach to ensuring integrity without the use of resource-intensive blockchain algorithms (Proof-of-Work) is successfully employed in secure logging systems for medical IoT devices [9, 10, 11] and in offline-oriented systems for humanitarian missions, such as fEMR [12, 13]. The architecture also adheres to the principles of Patient-Controlled Encryption (PCE) [14, 15], where key management is decentralized at the end-user level; this is recognized as an effective alternative to centralized KMS in environments with elevated risks of physical access.

Conclusions

The paper proposes a cryptographic architecture for a distributed medical system designed to operate under conditions of partial server autonomy and the absence of a centralized Key Management System. The combination of a static user key, a rotating daily server key, and a local administrator master key forms a hybrid security model that balances operational simplicity with secure emergency recovery capabilities.

The implemented hash-chain structure guarantees the immutability of record history, technically satisfying the integrity control requirements of the HIPAA Security Rule §164.312(c)(1). The multi-layer envelope encryption scheme aligns with GDPR (Art. 32) requirements for pseudonymization and encryption of personal data, minimizing leakage risks in the event of physical server compromise.

Future research will focus on refining the key hierarchy (introducing a KEK layer) while maintaining system usability, integrating hardware security tokens, and developing AI-driven mechanisms for automated anomaly detection in access logs to identify suspicious behavior patterns. .

Author Contributions

The authors' contributions are as follows: V. Kysil proposed the methodology for the decentralized doctor-centric architecture, designed the algorithms, and carried out the software implementation; T. Kysil supervised the study and determined the results validation strategy.

Declaration on the use of generative artificial intelligence tools

The authors acknowledge the use of artificial intelligence tools (specifically Large Language Models) for assistance in English language editing, translation, and formatting of the manuscript. The scientific concept, architectural design, implementation, and final text verification remain the sole responsibility of the authors.

References

1. I. Kuzminykh, B. Ghita, and S. Shiaeles, "Comparative analysis of cryptographic key management systems," *arXiv preprint arXiv:2109.09905*, 2021. DOI: https://doi.org/10.48550/arXiv.2109.09905
2. M. A. Al-Khasawneh *et al.*, "A secure blockchain framework for healthcare records management systems," *Healthc. Technol. Lett.*, vol. 11, no. 6, pp. 461–470, Oct. 2024. DOI: https://doi.org/10.1049/htl2.12092
3. A. Ullah *et al.*, "Toward blockchain based electronic health record management with fine grained attribute based encryption and decentralized storage mechanisms," *Sci. Rep.*, vol. 15, no. 1, Art. no. 34542, 2025. DOI: https://doi.org/10.1038/s41598-025-17875-5
4. N. U. A. Tahir *et al.*, "Blockchain-based healthcare records management framework: Enhancing security, privacy, and interoperability," *Technologies*, vol. 12, no. 9, Art. no. 168, Sep. 2024. DOI: https://doi.org/10.3390/technologies12090168
5. S. Saif *et al.*, "A secure data transmission framework for IoT enabled healthcare," *Heliyon*, vol. 10, no. 16, Art. no. e36269, Aug. 2024. DOI: https://doi.org/10.1016/j.heliyon.2024.e36269
6. A. Dubovitskaya *et al.*, "ACTION-EHR: Patient-centric blockchain-based electronic health record data management for cancer care," *J. Med. Internet Res.*, vol. 22, no. 8, Art. no. e13598, Aug. 2020. DOI: https://doi.org/10.2196/13598
7. H. Guo, W. Li, M. Nejad, and C.-C. Shen, "A hybrid blockchain-edge architecture for electronic health record management with attribute-based cryptographic mechanisms," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 2, pp. 1759–1774, Jun. 2023. DOI: https://doi.org/10.1109/TNSM.2022.3186006
8. O. V. Shmatko and S. S. Salnikov, "Model of a decentralized system for exchanging electronic medical records based on blockchain technology," *Control, Navigation and Communication Systems*, no. 2, pp. 152–162, 2024. [Online]. Available: https://journals.nupp.edu.ua/sunz/article/view/3373
9. E. Soriano and G. Guardiola, "SealFS: Storage-based tamper-evident logging," *Comput. Secur.*, vol. 108, Art. no. 102325, Sep. 2021. DOI: https://doi.org/10.1016/j.cose.2021.102325
10. D. G. Chakravarthy, R. Gopi, S. Murugan, and E. R. Joseph, "Enhancing confidentiality and access control in electronic health record systems using a hybrid hashing blockchain framework," *Sci. Rep.*, vol. 15, no. 1, Art. no. 30379, Aug. 2025. DOI: https://doi.org/10.1038/s41598-025-13831-5
11. H. Nguyen *et al.*, "Cloud-based secure logger for medical devices," in *Proc. 2017 IEEE Int. Conf. Connected Health: Appl., Syst. Eng. Technol. (CHASE)*, Philadelphia, PA, USA, 2017, pp. 89–98.
12. T. Brotherton *et al.*, "Development of an offline, open-source, electronic health record system for refugee care," *Front. Digit. Health*, vol. 4, Art. no. 847002, Mar. 2022. DOI: https://doi.org/10.3389/fdgth.2022.847002
13. M. Draugelis-Brown *et al.*, "Development and implementation of an electronic health record system for use in humanitarian emergencies, disaster response, and conflict zones," *PLOS Glob. Public Health*, vol. 5, no. 1, 2025.
14. J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient controlled encryption: Ensuring privacy of electronic medical records," in *Proc. 2009 ACM Workshop Cloud Comput. Secur. (CCSW '09)*, Chicago, IL, USA, 2009, pp. 103–114. DOI: https://doi.org/10.1145/1655008.1655024
15. J. Eom, D. H. Lee, and K. Lee, "Patient-controlled attribute-based encryption for secure electronic health records system," *J. Med. Syst.*, vol. 40, no. 12, Art. no. 253, Dec. 2016. DOI: https://doi.org/10.1007/s10916-016-0621-3

Volodymyr Kysil **Володимир Кисіль**	PHD Student of Computer Engineering & Information Systems Department, Khmelnytskyi National University https://orcid.org/0009-0003-9387-6609 e-mail: vovikusspambox@gmail.com	Аспірант кафедри комп'ютерної інженерії та інформаційних систем, Хмельницький національний університет
Tetiana Kysil **Тетяна Кисіль**	Candidate of Physical and Mathematical Sciences, Associate Professor of Computer Engineering & Information Systems Department, Khmenlnytskyi National University https://orcid.org/0000-0002-4094-3500 e-mail: kysil_tanya@ukr.net	Кандидат фізико-математичних наук, доцент кафедри комп'ютерної інженерії та інформаційних систем, Хмельницький національний університет