Patient-centric health data sovereignty: an approach using proxy re-encryption*

Bruno Rodrigues[1][0009−0001−2929−0836], Ivone Amorim[2**][0000−0001−6102−6165], Ivan Silva[2][0009−0009−8480−9352], and Alexandra Mendes[3][0000−0001−8060−5920]

[1] Faculty of Engineering, University of Porto, Porto, Portugal
up202103390@edu.fe.up.pt
[2] PORTIC – Porto Research, Technology & Innovation Center, Polytechnic of Porto, 4200-374 Porto, Portugal
{ivone.amorim,ivcps}@sc.ipp.pt
[3] Faculty of Engineering, University of Porto, Porto & HASLab/INESC TEC, Portugal
alexandra@archimendes.com

Abstract. The exponential growth in the digitisation of services implies the handling and storage of large volumes of data. Businesses and services see data sharing and crossing as an opportunity to improve and produce new business opportunities. The health sector is one area where this proves to be true, enabling better and more innovative treatments. Notwithstanding, this raises concerns regarding personal data being treated and processed. In this paper, we present a patient-centric platform for the secure sharing of health records by shifting the control over the data to the patient, therefore, providing a step further towards data sovereignty. Data sharing is performed only with the consent of the patient, allowing it to revoke access at any given time. Furthermore, we also provide a *break-glass* approach, resorting to Proxy Re-encryption (PRE) and the concept of a centralised trusted entity that possesses instant access to patients' medical records. Lastly, an analysis is made to assess the performance of the platform's key operations, and the impact that a PRE scheme has on those operations.

Keywords: data-sovereignty· cryptography · PRE· access delegation· e-health · PHR

1 Introduction

The ever growing digitisation of services that we use daily, as well as the increasing interest in data crossing and sharing to improve processes, services, and

* This work was partially supported by the Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (ERDF), within project "Cybers SeC IP" (NORTE-01-0145-FEDER-000044).

** Corresponding author

achieve new business opportunities, raises concerns regarding how data is handled and processed. In the healthcare sector, data sharing is not only beneficial, but also needed to provide the best care possible to the patients. However, this data is also highly sensitive, which requires special care. Several governmental measures have already been taken to improve and standardise the way in which data is shared, such as the European Data Governance Act [6], GDPR[4] and, more specifically, in personal health information, HIPAA[5] and HITECH[6]. These directives instigate a user-centric paradigm, granting individuals sovereignty over their data.

Several approaches have been proposed for ensuring security and privacy in e-health systems. Conventional encryption techniques based on symmetric and asymmetric encryption like AES and ECC are commonly used [4]. However, these techniques become problematic when data needs to be shared among multiple entities due to redundancy and computational burden [5]. Alternative approaches that incorporate policy constraints, such as Attribute-Based Encryption (ABE), are potential solutions [5], but they come with their own complexities and limitations, such as managing attribute-based keys and overriding policies in emergencies [7]. ABE also lacks the fine-grained access control necessary for a patient-centric sovereign approach.

Proxy Re-Encryption (PRE) is a cryptographic solution for secure data sharing without prior knowledge of the recipient. Unlike ABE, it does not rely on policies or attributes. PRE converts a ciphertext to a recipient's key without revealing the plaintext to the intermediary entity. It is particularly useful in semi-trusted cloud environments [17]. In e-health, PRE has already been used to securely share medical records [20, 23, 19, 26], including in emergency scenarios [19]. However, challenges remain in terms of revocability, computational effort, and safeguarding emergencies [26]. Existing solutions for emergency scenarios are limited and rely on assumptions that may impact efficiency and reliability.

In this context, it is necessary to develop a platform that addresses the aforementioned concerns. This includes enabling more control over the data by the patient while ensuring the safety of that data, even in semi-trusted environments. This contributes to the collaborative aspect of e-health and thus enables better treatments and advancements in the healthcare sector.

In this paper, we present a platform that leverages PRE to enhance health data sharing. Umbral's PRE [16] is used as the foundation for re-encryption processes, through which we achieve unidirectionality and non-interactivity, ensuring secure re-encryption from the patient to the data user (e.g., practitioners or health centres) without requiring the data user's private key. This approach centres on the expressed consent of the patient to data sharing, eliminating the need for prior identification of authorised parties — a drawback identified in previous solutions. Additionally, our platform offers revocability options, such

[4] https://data.europa.eu/eli/reg/2016/679/oj
[5] https://www.cdc.gov/phlp/publications/topic/hipaa.html
[6] https://www.hipaajournal.com/what-is-the-hitech-act/

as time-based access limits and patient-initiated access revocation. Importantly, the revocation of access does not require changes to the encrypted healthcare database, distinguishing our platform from the ones that rely on identity and attribute-based PRE schemes.

Furthermore, in the context of healthcare, it is crucial to ensure data sharing in emergency situations when explicit patient consent may not be possible. Our platform addresses this challenge by incorporating a trusted entity for data access when patient authorisation is infeasible.

In summary, our main contributions are:

- A patient-centric platform, that empowers patients with sovereign control over their health data, enabling granular access control and facilitating the sharing of health records only with explicit consent.
- Robust data protection using Umbral's PRE, ensuring secure and encrypted health data sharing without compromising the data user's private key.
- A robust access revocation mechanism that enables time-based access limits and supports manual revocation by the patient at any time and with immediate effect.
- A break-glass mechanism to ensure seamless emergency data access.

The remainder of this paper is organised as follows. Section 2 introduces basic concepts and definitions, as well as the classification and properties of PRE schemes. Furthermore, an analysis is made concerning the framework on which the access delegation mechanism is based. Section 3 presents the current picture of the PRE and the advancements regarding break-glass scenarios. Section 4 details the proposed solution and its implementation. Section 5 is concerned with the performance test, respective results, and discussion. Lastly, Section 6 presents the conclusions and future work.

2 Proxy Re-encryption

PRE is a cryptographic technique that enables a third-party entity, named proxy, to delegate access to encrypted data, without being able to infer the plaintext content of that data. This is achieved by transforming a ciphertext encrypted under one key into a ciphertext encrypted under a different key.

2.1 Syntax and basic definitions

Since PRE can be seen as a way to delegate decryption rights to a party, it is possible to categorise the different entities according to the delegation relation they possess with each other. The *delegator* is the entity that owns the data and delegates decryption rights. The *proxy* is the intermediary entity in the delegation process, which uses a Proxy Re-encryption Key (PRK) to transform the ciphertext encrypted under the delegator's public key into a ciphertext that can be decrypted only by using the delegatee's private key. Finally, the *delegatee* is the entity that accesses the information through delegation of decryption rights by the delegator.

Definition 1 (PRE). *A PRE scheme can be defined based on five different algorithms:*

- ***KeyGen*** *— On input of a security parameter n, the key generation algorithm KeyGen outputs a public/private key pair (pk_A, sk_A) for a given user A.*
- ***ReKey*** *— On input of a public/private key pair (pk_A, sk_A) for user A and a public/private key pair (pk_B, sk_B) for user B, a PRK $rk_{A \to B}$ is computed.*
- ***Encrypt*** *— Given the input of a public key pk_A and a message $m \in M$, the encryption algorithm outputs a ciphertext $c_A \in C_1$.*
- ***ReEncrypt*** *— On input of a ciphertext $c_A \in C_1$ and a PRK $rk_{A \to B}$, the re-encryption algorithm ReEncrypt transforms a ciphertext $c_A \in C_1$ into a ciphertext $c_B \in C_2$.*
- ***Decrypt*** *— Given a private key sk_A from user A and a ciphertext $c_A \in C_S$ ($S \in \{1, 2\}$) from user A, the same executes the decryption algorithm and outputs the original message $m \in M$.*

According to Qin et al. [18], a PRE scheme can be classified based on its abilities. For example, regarding its directionality, we say that the scheme is *unidirectional* if it enables the delegator's ciphertext to be re-encrypted into the delegatee's ciphertext but not vice versa. Otherwise, we call it *bidirectional*. The multi-use/single-use classification focuses on the number of times the PRK can be used to re-encrypt data. In *multi-use* schemes, the PRK can be utilised to perform several re-encryptions. In the case of a *single-use* scheme, the PRK can only be used to perform a single transformation. Interactivity dictates whether the re-encryption is computed using just the public key from the delegatee (*non-interactive* scheme) or both the public and private keys (*interactive* scheme). Depending on the scenario of utilisation, some properties may be more desirable than others.

Other authors classify PRE schemes according to their way of functioning [9, 10]. For example, an Identity-Based PRE (IB-PRE) scheme derives public keys from identity attributes (e.g. email). The messages are encrypted using an identity string from the delegatee. Attribute-Based PRE (AB-PRE) schemes allow transforming a ciphertext defined by a set of attributes or access policies into another ciphertext with a different set of attributes.

2.2 Umbral's PRE scheme

The Umbral PRE scheme is, in its essence, a threshold PRE scheme. This scheme features an Elliptic Curve Integrated Encryption Scheme (EICS-KEM) inspired in ANSI X9.63 [1] and proposes several improvements over the original PRE scheme proposed by Blaze et al. [3], namely unidirectionality, non-interactivity, and verifiability. It relies on the concept of semi-trusted proxies, also known as ursulas. Being a threshold PRE scheme, it splits the PRK according to shares. The *threshold* portion of the scheme dictates the minimum number of those shares required to decrypt the information.

Splitting the PRK across multiple proxies brings some benefits namely eliminating a single point of failure, in case of a malfunction or compromise of one of the proxies the PRK is still safeguarded.

The re-encryption processes in our platform are supported by pyUmbral [15], a Python-based implementation of Umbral.

Figure 1 presents an overview of the main processes and data flows involved in the Umbral PRE scheme. This system beholds seven main processes: *Encapsulation*, *Encryption*, *Generate PRK fragments*, *Re-encapsulation*, *Decapsulation*, and *Decryption*. These processes are supported by three major cryptographic methods: Key Encapsulation Mechanism (KEM), Data Encapsulation Mechanism (DEM), and Shamir Secret Sharing (SSS) [21].

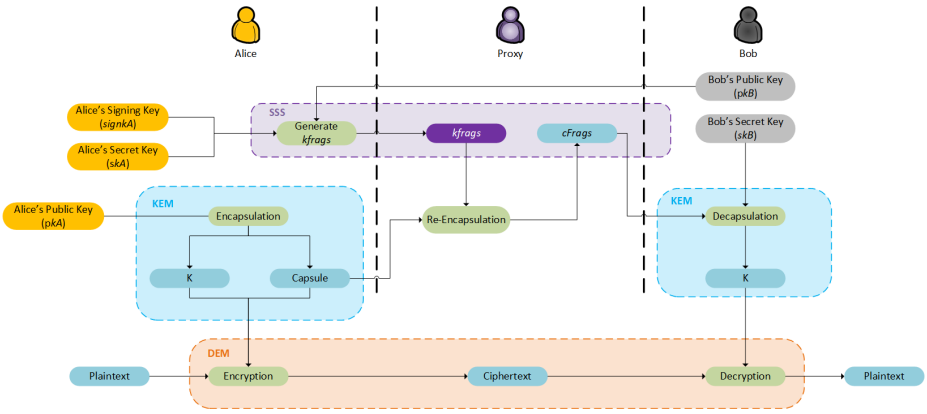

Fig. 1. Procedural overview of pyUmbral PRE scheme

The first step in this process is *Encapsulation*. This is achieved through the use of a Key Encapsulation Mechanism (KEM), in this case, a loosely inspired implementation of ECIES-KEM introduced by Shoup [22]. The KEM is fed with Alice's public key pk_A and outputs a symmetric key K and a *capsule*.

With the capsule and the symmetric key K, the *Encryption* process is performed using a Data Encapsulation Mechanism (DEM) which uses Authenticated Encryption with Additional Data (AEAD). This outputs a ciphertext encrypted with the symmetric key.

When the data is encrypted and stored in the cloud, in order for the access delegation to occur, there is a need to generate a PRK. This is performed by the *Generate PRK fragments* process resorting to the notions present in Shamir Secret Sharing [21], Alice's private key and signing key $signk_A$, and Bob's public key pk_B. This enables the generation of the PRK fragments or *kFrags*. The number of fragments is defined by the number of shares.

The *kFrags* are stored by the proxy for further use in the *Re-encapsulation* process. This process is responsible for generating the *cFrags* which enables Bob

to gain access to the file at a later stage. To generate the *cFrags* just the capsule and the *kFrags* are needed. This is due to the fact that this PRE scheme performs the re-encryption over the capsule.

Lastly, once Bob wants to retrieve a file, the *Decapsulation* process needs to happen. This process resorts to SSS in order to reconstruct the symmetric key k. To do so, Alice's public key, Alice's verifying key vk_A for signature verification of the *cFrags*, Bob's private key sk_B, and the *capsule* are needed. Through the use of a Key Derivation Function within the KEM, it is possible to derive the symmetric key K which together with the ciphertext is passed to the DEM. The DEM performs the *Decryption* process and outputs the plaintext content of the file that Bob can now use.

3 Related work

The notion of PRE made its first appearance in 1998 when Blaze et al. [3] introduced the concept of bidirectional and multi-use PRE. Several works have been published since then with new PRE schemes providing new functionalities and relying on different mathematical assumptions. For example, both Hanaoka et al. [8] and Kirshanova [11] proposed a unidirectional, single-use PRE scheme, but the first relies on threshold PKE, while the second is based on lattice-hardness problems. In 2015, Liang et al. [14] also proposed a unidirectional and single-use PRE scheme, which can be classified as attribute-based. Later, in 2017, Nuñez [16] presented a unidirectional, non-interactive, and verifiable PRE scheme which is threshold-based.

In the context of healthcare data sharing, PRE has also been widely explored. In fact, several works address security, privacy, and confidentiality when it comes to the design and implementation of e-health systems. However, there is still a lack of development concerning safeguarding emergency scenarios in the context of e-health systems [26]. Works that address this kind of scenario in its design, refer to this as break-glass approaches. In 2017, Au et al. [2] proposed a framework for the secure sharing of Personal Health Records (PHRs) that relies on attribute-based PRE and which addresses emergency scenarios. The break-glass capabilities are provided with ABE. In this scheme, the emergency department attribute is always appended to the policy that encrypts the patient PHR, thus providing instant access to the entity from the moment the same is uploaded. The problem with this approach, and in general with ABE approaches, is that they present some caveats, namely key management and resorting to other mechanisms in break-glass approaches. This is due to the fact that emergency normally means an exception to a policy and, thus, overriding that same policy might be a hefty task in some implementations. In 2019, Yang et al. [25] also proposed an approach that is based on an attribute-based PRE, and provided self-adaptive access control, meaning that the system can automatically adapt to normal or emergency situations. However, their break-glass mechanism resorts to a *password-based* paradigm. This approach raises some concerns, namely in the assumption that the individual that stores the password has the necessary

means to ensure its secrecy. More recently, in 2022, Li et al. [13] proposed a system for IoT sensors combining PRE and PKE with equality test, permitting searches under different public keys and secure data sharing. However, it does not discuss emergency situations. In the same year, Ren et al. [20], proposed a non-interactive, multi-use, certificateless PRE for sharing health data in a cloud environment. Even though their approach gives full control to the data owner, it has two important drawbacks, namely it is interactive and does not propose a break-glass mechanism. Also in 2022, Xue [24] published a secure data sharing and authorised Searchable framework for e-healthcare systems. This framework lies on a conditional and unidirectional PRE scheme with keyword search. It is also idealised for managing sensible data from medical wearable devices. This platform has some disadvantages namely regarding the PRK generation performance. Also, this work does not address emergency situations. Finally, in 2022, Li et al. [12] propose a framework which is also based on attribute-based PRE that features break-glass capabilities. However, it leaves open the possible solution for revocability. That being said, there is a need to develop a solution that can cope with all the aforementioned concerns and that contributes to a more reliable and robust break-glass approach.

4 Patient-centric health data sovereignty

In this section, we introduce the envisioned solution for a patient-centric platform that enables health data sovereignty through PRE. The subsequent section presents the architecture of the solution, followed by a description of the processes involved in the key operations for access delegation.

4.1 Proposed Solution

The proposed solution consists of four main nodes: the client, the resource server, the proxy server, and the authorization server, as depicted in Figure 2.

The client node hosts the client-side application developed with Next.js[7]. This client node communicates with the server nodes via Representation State Transfer (REST) and the Hypertext Transfer Protocol (HTTP). The business logic is divided between the resource and proxy server nodes. The resource server is based on the FastAPI framework[8] running in a Python environment. This server is trusted by the data delegator and it is responsible for assisting the client-side operations, namely feeding the data the client node needs to display the information to the user. The resource server node also performs some core operations such as the initial encryption and final decryption of the Electronic Health Record (EHR) stored in the database server node hosted in a cloud environment (MongoDB[9]) as well as the management of delegation requests (accept or decline). Some other complementary operations are also performed

[7] https://nextjs.org/
[8] https://fastapi.tiangolo.com/
[9] https://www.mongodb.com/

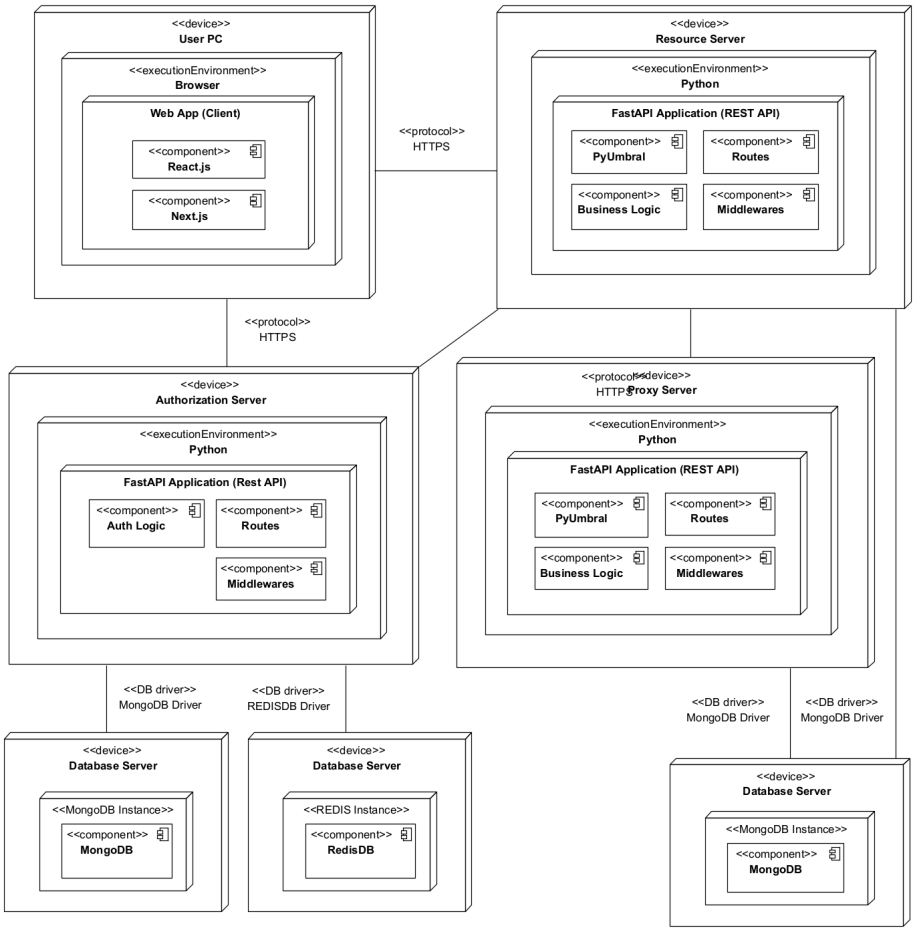

Fig. 2. Deployment diagram of the architecture

such as the generation of the PRK which is stored afterwards by the proxy server node, and the signature verification of the PRK fragments and capsule fragments.

The proxy server is solely responsible for the process of EHR delegation, being used for the re-encryption of the capsules and the storage of the PRK.

The authorisation server is responsible for performing the authentication of the different users of the platform as well as the issuing and claims verification of the authorisation tokens. These tokens are subsequently used to consume the APIs provided by the resource and proxy server nodes. This node is also associated with two persistence nodes. An in-memory database (REDIS[10] instance) for persisting and performing the lookup of the refresh tokens and a MongoDB

[10] https://redis.com/

instance for storing general purpose user information such as name, email, password, public and verifying keys and roles.

4.2 Authentication/Authorisation

The authorisation is performed by resorting to JSON Web Tokens (JWT) which are signed using HMAC SHA256. This ensures the tokens can not be tampered with or changed, thus enabling secure transmissions between parties.

The authentication flow comprises a traditional email/password authentication, where each user needs to provide a valid email and password. In case of successful authentication, a pair of tokens are issued (access/refresh token) containing some claims needed to support the client-side application. These claims follow the standards and restrictions defined in Request for Comments 7519[11]. Besides the pair of tokens, a Cross-site Request Forgery token is also sent for further protection in requests that require cookies. The refresh token is also sent in a cookie configured with the *secure* and *httpOnly* flags to ensure it is only transmitted through HTTPS and not available to JavaScript in case of a Cross-site Scripting vulnerability in the client-side application.

Since JWT tokens are self-contained, there is no natural way of revoking them. In order to tackle this problem anti-theft mitigation techniques were implemented: *refresh token rotation* and *token reuse detection*.

4.3 Access delegation scenario

Access delegation is the core problem tackled in this work. The next sections dissect the access delegation flow from the moment the file is uploaded by the patient to the moment the plaintext content is retrieved by the healthcare provider. For demonstration purposes, the step-by-step process between two entities, Alice (delegator) and Bob (delegatee), is presented.

Upload of an EHR The access delegation starts with the upload of an EHR by Alice. When Alice uploads a new EHR, which can be a Portable Document Format (PDF) or an image, the resource server encrypts the file using a symmetric key resulting from the encapsulation process and stores it together with the capsule, resulting from the *encapsulation* process, and an associated *userId*.

Another process that is also performed in this step and further detailed in Section 4.4 is the safeguarding of emergency situations. Besides the persistence of the file in the database, a PRK is also generated in order to provide access to the predefined trusted entity. This ensures that the trusted party possesses the means to access the file from the moment it is uploaded and that no extra input from the user is needed in this regard. This PRK is sent to the proxy for subsequent use.

Bob requests access to an EHR When Bob wants to access Alice's uploaded EHR, he needs to formalise his intentions by issuing a share request to the resource server containing the EHR's *resourceId*. In this step, the system checks

[11] https://datatracker.ietf.org/doc/html/rfc7519#section-4

if Bob is the owner of the EHR. This prevents a user from performing a share request to itself, something that violates the business rules of the platform since if the file is owned by a given user, it automatically has access to it and no share request is needed.

Once this validation is performed, and provided with the *resourceId*, the resource server generates a share request that includes the *resourceId*, the *delegatorId* and the *delegateeId*, as well as a *status* that is set to pending by default.

Alice answers the share request Now that Bob asked Alice for access to the EHR, Alice is now capable of answering the share request. Depending on Alice's answer, the execution flow might have two outcomes:

Accept scenario — In case of an acceptance, Alice needs to generate the PRK required to re-encrypt the capsule and further enable Bob to have access to the plaintext content of the EHR. To achieve such a feat, Alice requires her secret key and signing key pair, needed to verify the signature of the *kFrags* and *cFrags* at a later stage, as well as Bob's public key. Notice that just the public key is needed, due to the non-interactivity property of this PRE scheme. Lastly, since the underlying scheme of the access delegation mechanism is a threshold PRE scheme, there is also the need to provide a *threshold* which defines the minimum number of shares needed to decrypt the capsule and the number of *shares* which dictates the number of outputted PRK fragments. This last aforementioned operation outputs the *kFrags*, which are sent to the proxy along with a *shareId* binding the PRK to a specific share request. Both attributes are persisted by the proxy for further use once Bob retrieves the EHR.

The share request operation ends with the status update of the share request, which is defined as accepted, together with an arbitrary expiration date defined by Alice. This expiration date is optional, being possible to share an EHR indefinitely or temporarily, in which case the share request is automatically revoked through a cron job once that date is transposed. This ensures the time-based access delegation aspect that this work contributes to.

Decline scenario — In case Alice declines the share request, the status is updated accordingly and no other action is performed.

Bob retrieves the EHR Now that Alice explicitly delegated access to the EHR, Bob is now capable of retrieving it. To do so, Bob performs a request to the resource server, which requires Bob's secret key and the *resourceId*, which uniquely identifies the EHR. A file ownership verification is also performed since the decryption steps are different for a delegator and a delegatee, where the former does not have the need to re-encrypt the *capsule*.

As stated previously, ownership trails different paths regarding execution flow. With that said, the following can happen whether the user is or not a data owner.

Data owner — In case the user that requests the file is a data owner, a hybrid decryption scenario is performed where the data owner's private key is used to decapsulate the private key used to initially encrypt the file being retrieved, thus no re-encryption takes place.

Not a data owner — If the user is not the data owner, meaning they are a delegatee, a collaborative operation between the resource and proxy servers is required to take place. For this specific scenario, Bob needs to ask the proxy to re-encrypt the capsule using the previously generated PRK. To that purpose, the resource server retrieves the EHR details and sends the capsule to the proxy server. The proxy, equipped with the capsule and the PRK fragments *kFrags*, performs the *re-encapsulation* process outputting the *cFrags*. These *cFrags* are sent back to the resource server, which validates their signature through Alice's verifying key. Once the capsule fragments are validated, Bob decrypts the file by opening the capsule. This last step encompasses Bob's private key, Alice's verifying key, and the verified *cFrags*. With the plaintext content of the EHR, Bob is now capable of accessing the information.

Some important remarks to highlight are that the secret key used in the sharing process is never shared with the intermediary entity or proxy, making it semi-trusted. Additionally, the proxy only stores the PRK, which alone does not grant it the capability to decrypt the file. Furthermore, even if the stored information such as the capsule, PRK, and ciphertext were to be leaked from the database, the safety and integrity of the EHRs would still be preserved, as they are not sufficient for decrypting the EHRs.

4.4 Break-glass approach

Safeguarding emergency scenarios is of paramount importance in a health-related platform. Therefore, we adopted an approach that features a central trustworthy entity responsible for managing the authorisation in emergency scenarios. This trustworthy entity is seen as a government entity that is responsible for managing such issues and has full access to the files submitted in the platform.

The implementation is similar to what is described in Section 4.3 regarding Alice accepting the share request. However, in this case, there is no explicit acceptance of the share request. When an EHR is uploaded, the trusted entity user is retrieved from the database and a PRK is generated. An accepted share request is automatically created for the trusted entity, which links the PRK to the share request between the patient and the trusted entity.

Regarding the process of retrieving the EHR, it follows a similar procedure as depicted in Section 4.3. Just like in a regular file retrieval, since the share request is automatically accepted and the proxy possesses the PRK, the trusted entity requests the proxy to re-encrypt the capsules, enabling the final decryption to take place.

This approach vastly reduces the dependency on external actors, increasing the reliability and availability of the idealised break-glass approach. Having a dedicated entity for this purpose enables instant and swift access to the information if needed.

5 Performance analysis

In this section, we present the performance tests conducted to evaluate our platform. Given the common concerns of limited hardware infrastructures and sub-optimal conditions in governmental adoption cases, it is important to assess the responsiveness of the key operations offered by the platform. Our main goal is to quantitatively analyze the performance of the most computationally intensive operations and assess the impact of the PRE scheme. As there are no specific regulations, indications, or suggestions regarding performance for this type of platform, our tests are purely quantitative and based on known factors and conditions.

The performance tests were carried out on a deployed version of the platform, hosted in Microsoft Azure using a Free F1 tier running Linux and Python 3.10. While these specifications may be basic, they are sufficient to simulate a sub-optimal environment. In real-world scenarios, it is common for governments to have financial restrictions, making it likely that the platform would be deployed on infrastructure with modest specifications. The tests were conducted using Apache JMeter as the tool of choice.

In the rest of this section, we present the results related to the three most crucial operations of the platform and which involve the use of PRE: file upload, accepting a share request, and file retrieval. Additionally, a brief analysis of the results is also presented.

File upload The performance tests depicted in this section aim to evaluate how the different file sizes impact the upload performance of files.

Since the size of EHRs depends on various factors, such as the patient's medical history, the image resolution of the machines used for exams, and the content of the file itself, determining an average file size becomes challenging. Therefore, we conducted our experiments using two different file sizes: 1MB and 10MB.

Figure 3 illustrates the results obtained from a series of twenty runs performed for each file size.

It can be observed that a tenfold increase in file size reflected an average increase of 2715 *milliseconds*(ms) when comparing file sizes of 1MB and 10MB respectively. The former took an average of 1154 *ms* and the latter an average of 3870 *ms*.

Despite a time of almost four *seconds*, and considering this is not an ideal response time for a REST API, it should be taken into account the complexity of the operations performed. Since this is not a critical operation when it comes to performance, these values are acceptable.

Accepting a share request The acceptance of a share request is a key operation in the platform described in this paper. Although its performance does not possess a high impact on the efficiency of the platform, it does provide valuable information regarding the PRE process. In this operation, the PRK is generated and sent to the proxy for persistence purposes. Notice that, in this case, there was no need to perform the tests for both file sizes since the PRK generation only depends on cryptographic keys.

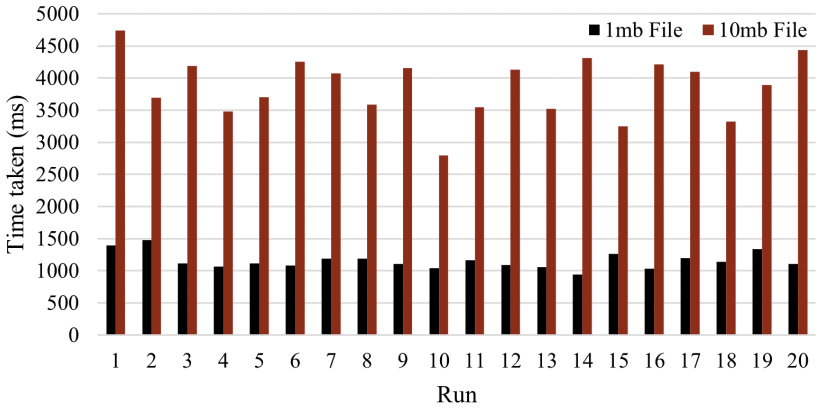

Fig. 3. Performance Tests - File Size Uploads Bar Chart

Regarding the results of these tests, the average time obtained in 20 runs was 869 *ms*. This quick response was expected since the generation of the PRK fragments is a relatively simple operation that depends on the cryptographic key from both ends, the signature, and the number of shares. Additionally, there was not a significant variation among the twenty runs that were performed. This is supported by the low standard deviation of just 188 *ms*.

File retrieval This set of tests aims to assess the impact of file sizes and the use of PRE on a file retrieval scenario. The tests were conducted for both regular decryption and PRE decryption. To evaluate the impact of file sizes, the tests were performed for both 1MB and 10MB file sizes.

Moving on to the obtained results (Figure 4), a 1MB file took an average of 903 *ms* to be retrieved while the 10MB one took an average of 2529 *ms*. Regarding file retrieval with PRE, the 1MB file took an average of 1245 *ms* and 2877 *ms* for the 10MB file.

We have also evaluated the impact of re-encryption on file retrieval operations (Figure 5) by directly measuring the difference between regular decryption and PRE for each file size. This resulted in an average difference of 342 *ms* for the 1MB file and 348 *ms* for the other one.

The results of our tests indicate that there is a similar average difference between regular and PRE decryption for both file sizes. This similarity can be attributed to the fact that the re-encryption process only affects the capsule, not the actual file. Since the sizes of the capsule and cryptographic keys are similar in both scenarios, it is expected that the results would be similar as well. The file size does not significantly impact the re-encryption of the capsule, but rather affects the overhead associated with fetching the file from the database and delivering it in the response.

Regarding the obtained results, they were deemed satisfactory since most operations do not possess restrictive requirements when it comes to performance.

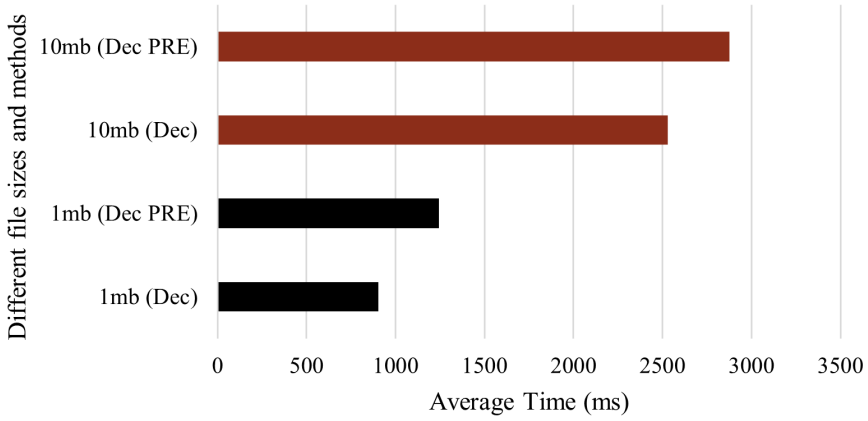

Fig. 4. Performance Tests - Average Time Taken for File Retrieval

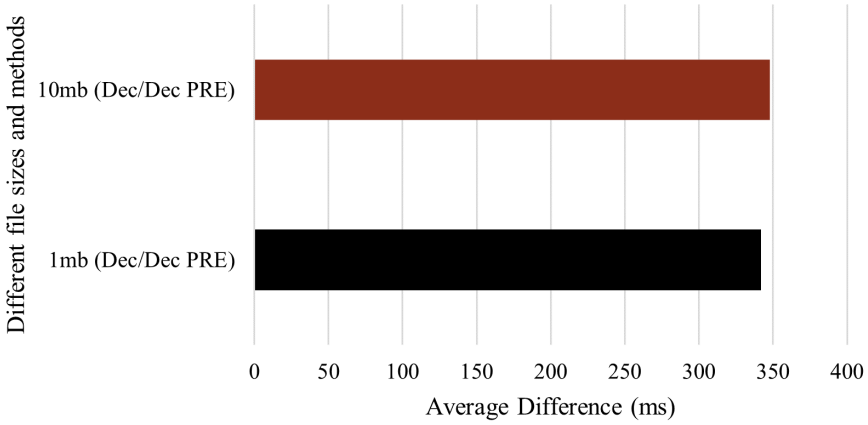

Fig. 5. Performance Tests - Average Impact of PRE in Same Sized Files

Regarding more critical operations such as file retrieval, considering the computational effort and infrastructure complexity required to ensure full correctness with the underlying threshold PRE scheme, the results were deemed satisfactory. It is important to note that these tests were conducted in a shared infrastructure with modest specifications. Thus, it was not possible to control the current workload of the servers during the tests, which may have impacted negatively the aforementioned results.

6 Conclusion

In this paper, we present a PRE-based platform for the secure sharing of e-health, considering a sovereign approach focused on the patient. This approach is achieved by ensuring that the patient's data is only shared with their explicit consent. Furthermore, it also enables robust revocability by the patient, without requiring updates on the encrypted EHR database, further contributing to a user-centric approach. Non-interactivity is also a key characteristic of our platform, which does not require sharing user's private key for the re-encryption process to occur. Another key achievement of our work is the proposed break-glass mechanism. Since some implementations fall short in terms of revocability, and only a few contemplate PRE in emergency scenarios, our solution uses a central trusted entity to which the proxy delegates access from the moment the EHR is uploaded to the platform. This eliminates the need to trust external actors in the system, increasing reliability and allowing swift access to the information in critical situations. There are other key characteristics of our platform worth highlighting. Firstly, it uses symmetric encryption to encrypt the EHR, which is faster than PKE. Secondly, the re-encryption process is performed over the capsule, which tends to have a much smaller size compared to a PHR. The tests that were conducted and our results show that the most demanding task is the upload of the EHR, as expected, because it requires the encapsulation process to occur and the encryption of the EHR. However, the re-encryption process does not show a significant increase when the size of the uploaded files increases. This is because the re-encryption does not involve the EHR. Our platform provides a solution to the sharing of medical data that incorporates key functionalities not covered together in previous literature, such as unidirectionality, non-interactivity, revocability, and a mechanism to deal with emergency situations. This solution contributes to the collaborative aspect of e-health and enables better, and more informed treatments supported by the increased exchange of information between providers.

Regarding future work, it would be beneficial to extend the architecture to accommodate multiple proxies instead of using just one. This could be achieved by utilising a decentralised approach where the proxies work together to re-encrypt the capsules, thus enabling all the benefits that a threshold-based scheme has to offer. Furthermore, additional tests could be performed using different environments and network conditions to cover more use case scenarios.

References

1. American National Standards Institute (ANSI) X9.F1 subcommittee. ANSI X9.63 Public key cryptography for the Financial Services Industry: Elliptic curve key agreement and key transport schemes (Jul 5, 1998), working draft version 2.0
2. Au, M.H., Yuen, T.H., Liu, J.K., Susilo, W., Huang, X., Xiang, Y., Jiang, Z.L.: A general framework for secure sharing of personal health records in cloud system. Journal of Computer and System Sciences **90**, 46–62 (2017). https://doi.org/https://doi.org/10.1016/j.jcss.2017.03.002

3. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. Advances in Cryptology—EUROCRYPT'98 pp. 127–144 (1998)
4. Edemacu, K., Park, H.K., Jang, B., Kim, J.W.: Privacy provision in collaborative ehealth with attribute-based encryption: Survey, challenges and future directions. IEEE Access **7**, 89614–89636 (2019). https://doi.org/10.1109/ACCESS.2019.2925390
5. ENISA: Engineering personal data sharing - emerging use cases and technologies (January 2023), https://www.enisa.europa.eu/publications/engineering-personal-data-sharing
6. European Parliament, C.o.t.E.U.: Regulation (EU) 2022/868 of the European Parliament and of the Council of 30 May 2022 on European data governance and amending Regulation (EU) 2018/1724 (Data Governance Act) (Text with EEA relevance) (May 2022), http://data.europa.eu/eli/reg/2022/868/oj/eng, legislative Body: CONSIL, EP
7. Fernández-Alemán, J.L., Señor, I.C., Ángel Oliver Lozoya, P., Toval, A.: Security and privacy in electronic health records: A systematic literature review. Journal of Biomedical Informatics **46**(3), 541–562 (2013). https://doi.org/https://doi.org/10.1016/j.jbi.2012.12.003
8. Hanaoka, G., Kawai, Y., Kunihiro, N., Matsuda, T., Weng, J., Zhang, R., Zhao, Y.: Generic construction of chosen ciphertext secure proxy re-encryption. In: Dunkelman, O. (ed.) Topics in Cryptology – CT-RSA 2012. pp. 349–364. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
9. Inbarani, W.S., Shenbagamoorthy, G., Kumar Charlie, C.: Proxy re-encryption schemes for data storage security in cloud- a survey. International Journal Of Engineering Research & Technology (IJERT) **02**(01) (2013)
10. Khan, F.: A comparison of proxy re-encryption schemes – a survey. International Journal of Computer Science and Information Security (IJCSIS) **14**, 392–397 (07 2016)
11. Kirshanova, E.: Proxy re-encryption from lattices. In: PKC. pp. 77–94. Springer (2014). https://doi.org/10.1007/978-3-642-54631-0_5
12. Li, M., Yu, S., Ren, K., Lou, W.: Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings. In: Jajodia, S., Zhou, J. (eds.) Security and Privacy in Communication Networks. pp. 89–106. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
13. Li, W., Jin, C., Kumari, S., Xiong, H., Kumar, S.: Proxy re-encryption with equality test for secure data sharing in internet of things-based healthcare systems: Na. Transactions on Emerging Telecommunications Technologies **33**, e3986 (06 2020). https://doi.org/10.1002/ett.3986
14. Liang, K., Fang, L., Wong, D., Susilo, W.: A ciphertext-policy attribute-based proxy re-encryption scheme for data sharing in public clouds: A cp-abpre for data sharing in public clouds. Concurrency and Computation: Practice and Experience **27** (10 2014). https://doi.org/10.1002/cpe.3397
15. NuCypher: pyumbral. https://github.com/nucypher/pyumbral (2018)
16. Nuñez, D.: Umbral: A threshold proxy re-encryption scheme (2017), https://raw.githubusercontent.com/nucypher/umbral-doc/master/umbral-doc.pdf
17. Nuñez, D., Agudo, I., Lopez, J.: Proxy re-encryption: Analysis of constructions and its application to secure access delegation. Journal of Network and Computer Applications **87**, 193–209 (2017). https://doi.org/https://doi.org/10.1016/j.jnca.2017.03.005

18. Qin, Z., Xiong, H., Wu, S., Batamuliza, J.: A survey of proxy re-encryption for secure data sharing in cloud computing. IEEE Transactions on Services Computing pp. 1–1 (2016). https://doi.org/10.1109/TSC.2016.2551238
19. Rabieh, K., Akkaya, K., Karabiyik, U., Qamruddin, J.: A secure and cloud-based medical records access scheme for on-road emergencies. In: 2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC). pp. 1–8 (2018). https://doi.org/10.1109/CCNC.2018.8319175
20. Ren, C., Dong, X., Shen, J., Cao, Z., Zhou, Y.: Clap-pre: Certificateless autonomous path proxy re-encryption for data sharing in the cloud. Applied Sciences **12**(9) (2022). https://doi.org/10.3390/app12094353
21. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (nov 1979). https://doi.org/10.1145/359168.359176
22. Shoup, V.: A proposal for an iso standard for public key encryption. Cryptology ePrint Archive, Paper 2001/112 (2001), https://eprint.iacr.org/2001/112
23. Thilakanathan, D., Chen, S., Nepal, S., Calvo, R., Alem, L.: A platform for secure monitoring and sharing of generic health data in the cloud. Future Generation Computer Systems **35**, 102–113 (2014), special Section: Integration of Cloud Computing and Body Sensor Networks; Guest Editors: Giancarlo Fortino and Mukaddim Pathan
24. Xue, L.: Dsas: A secure data sharing and authorized searchable framework for e-healthcare system. IEEE Access **10**, 30779–30791 (2022). https://doi.org/10.1109/ACCESS.2022.3153120
25. Yang, Y., Zheng, X., Guo, W., Liu, X., Chang, V.: Privacy-preserving smart iot-based healthcare big data storage and self-adaptive access control system. Information Sciences **479**, 567–592 (2019). https://doi.org/https://doi.org/10.1016/j.ins.2018.02.005
26. Yüksel, B., Küpçü, A., Öznur Özkasap: Research issues for privacy and security of electronic health services. Future Generation Computer Systems **68**, 1–13 (2017). https://doi.org/https://doi.org/10.1016/j.future.2016.08.011