KeRePre

A Key-Redistribution Proxy Re-encryption for DLT-based data sharing

Fadi Barbàra, Mirko Zichichi, Stefano Ferretti, Claudio Schifanella

Outline

- Problem Description and Goals
- Background
- Solution presentation
- Conclusions

Introduction

Trust in cloud-based data services

- **Cloud-based data services involve storing sensitive data on remote servers** operated by third-party service providers.

- The problem of trust arises when **users have to rely on the service providers to manage and access their data** securely and responsibly, which can lead to **privacy and security risks**.

Limitation of decentralization

- In a **decentralized data service, the problem may be exacerbated** as managers and operators are **usually pseudonymous**, making it difficult to establish trust.

- Data breaches and **leaks in cloud-based data services have happened in the past**, which highlights the importance of ensuring the security and privacy of sensitive data.

Limitations of data encryption

- **Data encryption enhances data security** by converting plaintext into ciphertext, so that it **cannot be read** or accessed by **unauthorized parties** without the decryption key.

- However, encryption can **hinder data sharing**, collaboration and searching because the encrypted data can only be decrypted by those who possess the decryption key.

Data Protection

- Data protection: **ensure the confidentiality, integrity, and availability of sensitive data.**

- **Data protection regulations** such as GDPR have also been introduced to ensure that organizations handle personal data responsibly and securely

- The **lack of centralized control** in decentralized data services can make it **harder to enforce compliance** with data protection regulations, such as GDPR.

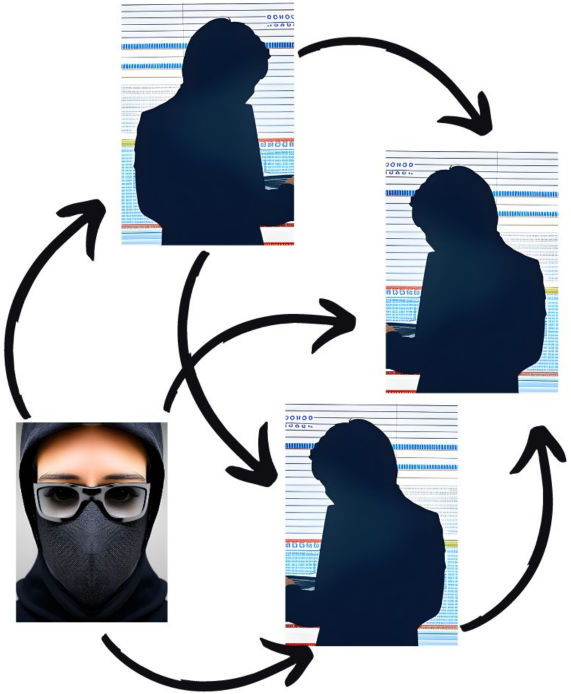
KeRePre Overview

- KeRePRE is a **decentralized and encrypted data-service** that aims to **enable secure data-sharing in decentralized file storage systems**.
- KeRePRE is a **three-components** system:
    a. a **threshold proxy re-encryption scheme** allows **Authorization Servers (AS)** to **transform ciphertext** encrypted under one public key into ciphertext encrypted under a different public key.
    b. a **key redistribution mechanism** allows addition or removing AS in a decentralized and trustless way.
    c. an **access control list (ACL) stored on a distributed ledger technology (DLT)** which can be read-only accessed by the AS. The ACL **lists the public keys of authorized users**.
        - In particular **AS can not read the data**

Background and Related Work

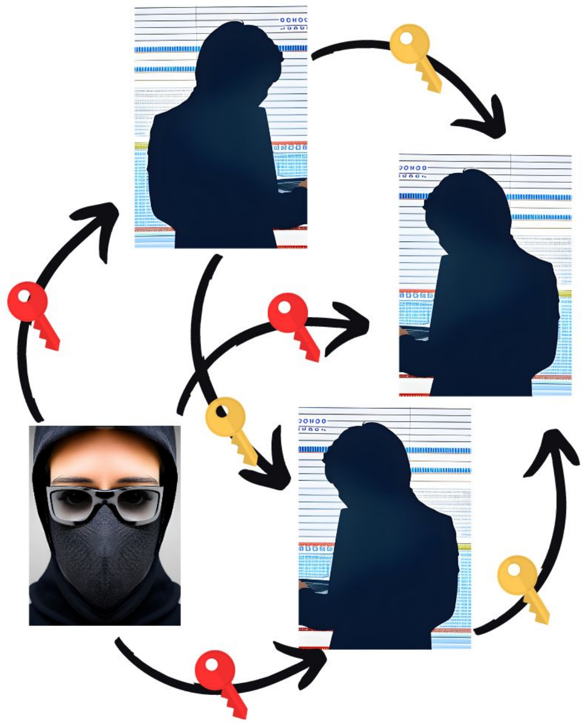
Shamir Secret Sharing

- A ***(t,n)*** Shamir Secret Sharing Protocol (1979) is a method of **securely sharing a secret among a group of people**.
- The secret is **divided into multiple shares (*n*),** each of which is distributed to a different participant.
- The protocol requires a **minimum number of participants (threshold *t*) to combine their shares** in order to reconstruct the secret
    - beside that number they may be *faulty*
- The protocol uses **polynomial interpolation** to generate the shares and to reconstruct the secret.
- The security of the protocol is based on the fact that it is **computationally infeasible to reconstruct the secret from less than the required number of shares**.



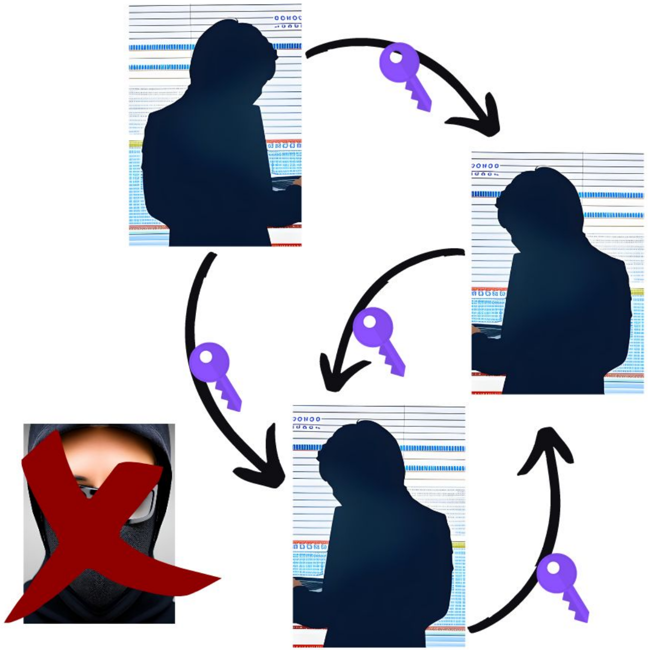
Threshold Cryptosystem

- A Threshold Cryptosystem (generally based on Shamir Secret Sharing Protocol) is a type of cryptography that allows a group of participants to jointly generate and use a cryptographic key.
    1. The **secret key is divided into shares** and distributed among participants.
    2. Each participant holds a share of the secret key, and the **key can only be reconstructed by combining** a minimum number of shares as determined by a threshold value.
- Threshold Cryptosystems provide better security as **the secret key is not held in its entirety by any single participant,** making it harder for any individual to compromise the key.



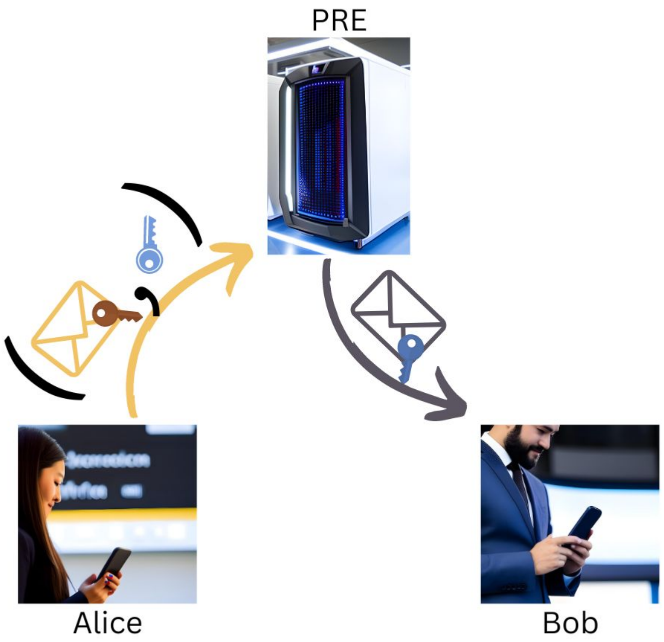
Key Redistribution

- Key Redistribution Mechanism is a cryptographic technique that enables the **re-distribution of a shared secret key** among a new set of participants.
- This mechanism can be used when **a subset of the original participants** of a shared secret key are **no longer available or trustworthy**, or when new participants need to be added.
- The key redistribution process typically involves the remaining participants distributing **new shares derived from the old ones of the same key** to the new or remaining set of participants.
- The new or remaining participants can then **use the new shares to reconstruct the secret key.**



Proxy Re-Encryption

- Proxy Re-Encryption (1998) is a type of public-key cryptography that allows a **third-party proxy to transform ciphertext encrypted under one public key into ciphertext encrypted under a different public key, without learning anything about the plaintext**.
    1. The **data owner first encrypts the data under their own public key** and then re-encrypts it under the public key of each recipient using a proxy.
    2. **The proxy is given a re-encryption key** by the data owner, which specifies the transformation to be applied to the ciphertext.
    3. The **recipient can then decrypt the re-encrypted ciphertext** using their own private key, without requiring access to the data owner's private key.



Proxy Re-Encryption Problems

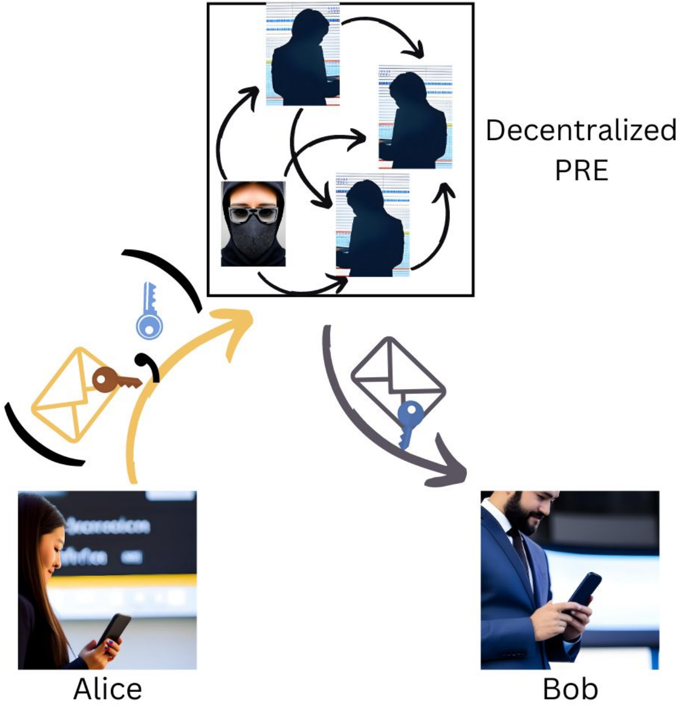
The Problems

1. **Proxy Re-Encryptors are centralized**, so the usual problem of centralized services do apply

The Problems

1. **Proxy Re-Encryptors are centralized**, so the usual problem of centralized services do apply

2. You may decentralize the PRE (threshold PRE), as in *Umbral*, **but what if more than *threshold* are malicious**?

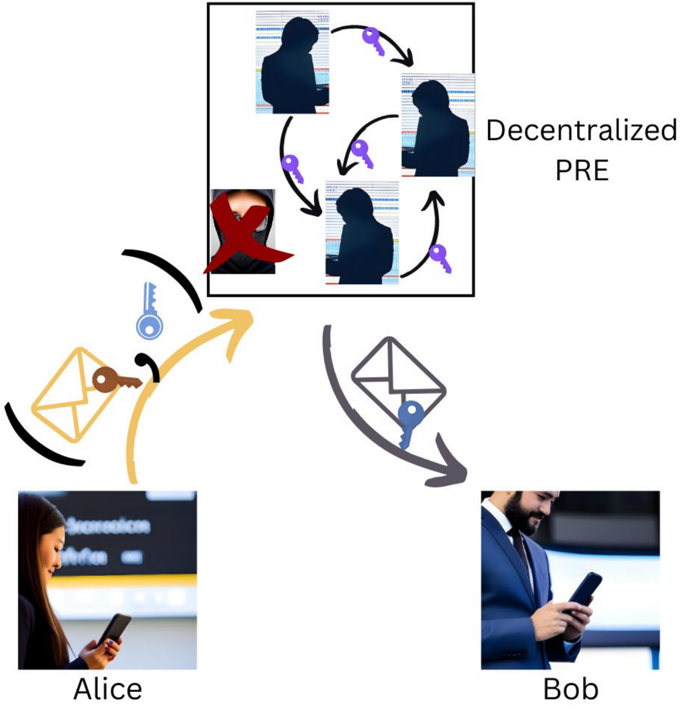


The Solution

1. **Proxy Re-Encryptors are centralized**, so the usual problem of centralized services do apply

2. You may decentralize the PRE (threshold PRE), as in *Umbral*, **but what if more than *threshold* are malicious**?

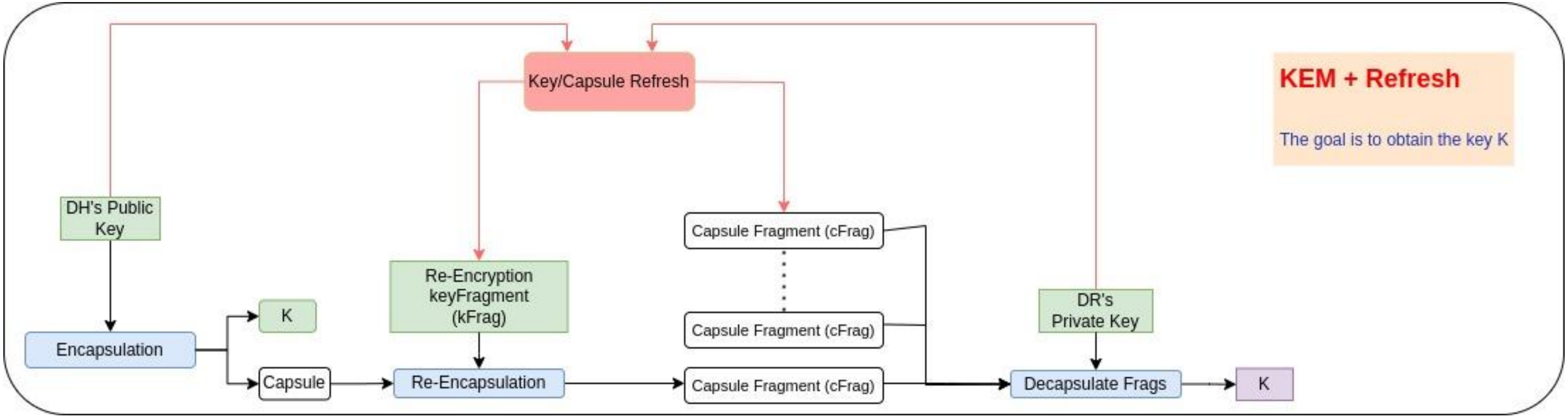
3. We can do a Key Re-Distribution and **remove the faulty** node or **add a new one (KeRePre)**

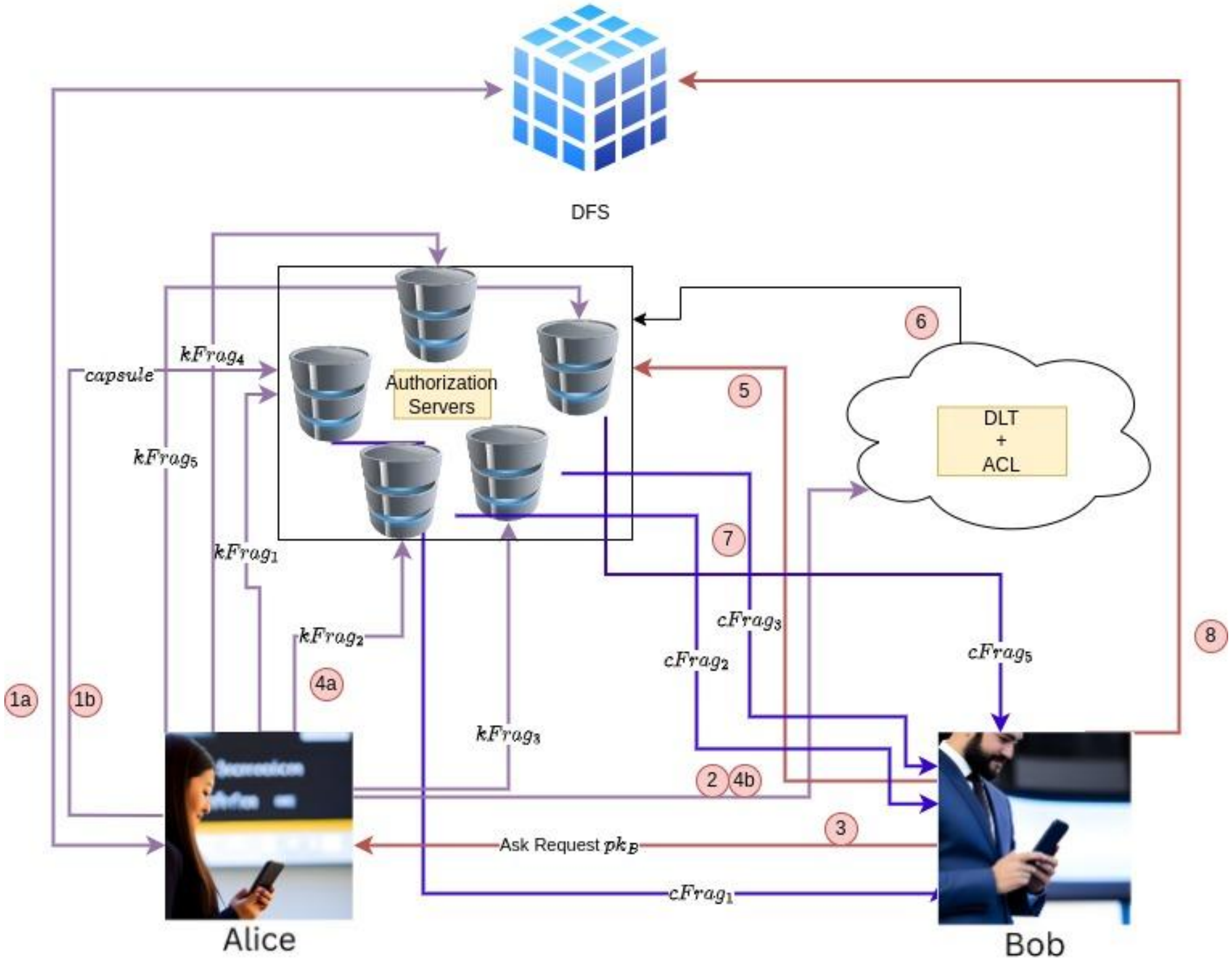


KeRePRE Architecture

Technical Terms

- kFrag: shares of the re-encryption keys
- capsule: piece of data related to file's encryption-key
- cFrag: share of the capsule





DFS
Authorization Servers
DLT + ACL
capsule
$kFrag_4$
$kFrag_5$
$kFrag_1$
$kFrag_2$
$kFrag_3$
$cFrag_1$
$cFrag_2$
$cFrag_3$
$cFrag_5$
Ask Request pk_B
1a
1b
2
3
4a
4b
5
6
7
8
Alice
Bob

The Solution

1. KeRePre **mitigates the centralization problems** of PRE: real-world ready threshold system thanks to key-redistribution mechanism

2. KeRePre **improves the data sharing process**: data can be accessed seamlessly thanks to DFS + TPRE

3. KeRePre **is compliant with data-protection regulations**: DLT-enforced ACL means no unauthorized user (especially AS) can not see the data

Conclusion

Conclusions

- KeRePre proposes a new method for **achieving decentralized and encrypted data sharing** that effectively **addresses the problems** discussed in the Introduction.
- We have extended the Umbral project and made it work in a data-sharing context.
- Our approach utilizes **key re-distribution** to effectively **solve scaling issues** and **mitigate the risks** posed by rogue/malicious nodes.