Personal Data Access Control Through Distributed Authorization

Mirko Zichichi*†§, Stefano Ferretti‡, Gabriele D'Angelo§, Víctor Rodríguez-Doncel†
†Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
mirko.zichichi@upm.es, vrodriguez@fi.upm.es
‡Department of Pure and Applied Sciences, University of Urbino "Carlo Bo", Italy
stefano.ferretti@uniurb.it
§Department of Computer Science and Engineering, University of Bologna, Italy
g.dangelo@unibo.it

***Abstract*—This paper presents an architecture of a Personal Information Management System, in which individuals can define the access to their personal data by means of smart contracts. These smart contracts, running on the Ethereum blockchain, implement access control lists and grant immutability, traceability and verifiability of the references to personal data, which is stored itself in a (possibly distributed) file system. A distributed authorization mechanism is devised, where trust from multiple network nodes is necessary to grant the access to the data. To this aim, two possible alternatives are described: a Secret Sharing scheme and Threshold Proxy Re-Encryption scheme. The performance of these alternatives is experimentally compared in terms of execution time. Threshold Proxy Re-Encryption appears to be faster in different scenarios, in particular when increasing message size, number of nodes and the threshold value, i.e. number of nodes needed to grant the data disclosure.**

I. Introduction

The transformation introduced by digital technologies has had (and is having) a significant impact on economy and society. Data is at the heart of this transformation and individuals are the main sources generating more and more of it. There is an urgent need to place (again) individuals at the center and to relieve the absence of technical instruments and standards that make the exercise of one's rights simple and not excessively burdensome [1], [2]. The EU's GDPR [1] helps to promote this vision and at the same time seeks to pave the way for open data spaces for the social and economic good [2].

Our aim is to seek such a technology by enabling users with the sovereignty over their data, while guaranteeing its confidentiality. In our view, the data owner can define access by limiting the scope of data utility, delegating these privileges or giving up ownership completely, without the need to rely on (un)trusted entities to facilitate this task. The development of a Personal Information Management System (PIMS) [3] that fulfils these goals can be based on a distributed software architecture, where each individual is associated to a digital space containing personal data. This space will be used to attend the data access requests coming from data providers and data consumers. Distributed Ledger Technologies (DLT) and Decentralized File Storages (DFS) combination provides a range of features suitable for data management and sharing, such as transparency, immutability and reliability [2], [3].

The contribution of the paper is the following. First, we propose an architecture for PIMS that, based on the use of DLTs, smart contracts and (D)FS enable to manage data access on the basis of multiple entities whose role is to provide mutual trust between the parties. Second, we employ two specific distributed authorization mechanisms, i.e. Secret Sharing (SS) and Threshold Proxy Re-Encryption (PRE) that allow for a decentralized distribution of personal data, yet guaranteeing data sovereignty to users, confidentiality and a secure access control. Third, we present an experimental evaluation of these considered schemes, showing that PRE is faster in different scenarios, although it has the drawback of requiring that the user generates a re-encryption key each new data consumer.

The remainder of this paper is organized as follows. Section II presents the background concepts and Section III describes related approaches. Section IV specifies the system architecture. Performance is evaluated in Section V before conclusions in Section VI.

II. Background

A. Distributed Ledger Technologies and Smart Contracts

DLTs provide a data ledger that guarantees the immutable persistence of the data, thereby ensuring untampered data availability. DLTs were born with the purpose of shifting trust from a human intermediary, who manages a transaction between two parties, to a protocol that allows two or more parties to conduct transactions directly. The resistance to manipulation makes DLTs able to support smart contracts. These consist of an immutable set of instructions that are executed deterministically by several participants in a network, who receive the same inputs and then perform a calculation which leads to the same outputs. When the issuer of a smart contract broadcasts it on the DLT network – and he is also

*This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie International Training Network European Joint Doctorate grant agreement No 814177 Law, Science and Technology Joint Doctorate - RIoE.

[1]Council of European Union, Regulation 2016/679 - directive 95/46

[2]European Commission, COM(2020) 66, "A European strategy for data"

[3]European Data Protection Supervisors, Opinion 9/2016, "EDPS Opinion on Personal Information Management Systems"

978-1-7281-8326-8/20/31.00

sure that the implemented behaviour is correct (e.g. through code auditing) – then the transactions originating from such a contract do not require the presence of a third party to be validated. In Ethereum [4], every process is completely traced and permanently stored in the blockchain because the smart contract computation is executed by all network participants.

B. Access Control Mechanisms

The objective of access control systems is to regulate access to system resources by enforcing permissions on the basis of a set of system policies for determining who can access information. Currently, most of them are based on a centralized controller that has the authority to access the data and, therefore, entails the risk of a single point of failure and, above all, privacy leakage [5]. In this paper we study techniques that are usually not strictly related to access control mechanisms, but that can offer a better privacy guarantee.

1) Secret Sharing: A sophisticated cryptographic technique that allows providing privacy to users for their data consists in Secret Sharing (SS), firstly proposed by Shamir [6] and Blakley [7]. This (t, n)-threshold scheme is used to share a secret between a set of n participants, with the possibility to reconstruct the secret using any subset of t (with $t \leq n$) or more shares, but no subset of less than t. By employing this in a network of (mostly honest) nodes, privacy is provided to a user that is sharing the secret, since none of the nodes can reconstruct the secret without the help of other $t - 1$ nodes.

2) Proxy Re-Encryption: PRE [8], is a type of public-key encryption, where a semi-trusted proxy entity transforms a ciphertext c, encrypted with a public key pk_1, into a ciphertext decryptable with a private key sk_2, without learning anything about the underlying plaintext. This is possible using a re-encryption key rk_{1-2} generated by the data owner who has the key pair (pk_1, sk_1) and and that divulges (to the proxy) the authorization of access to the plaintext to a data consumer holding the keypair (pk_2, sk_2). Among many schemes, in single-use uni-directional proxy re-encryption, the re-encryption function is one way.

III. Related Work

In the last years, many research efforts on access control have been done to securely store, share and transmit data while ensuring its integrity, validity and authenticity. Centralized controller issues may be addressed by adapting a solution based on a DLT for the verification of access permissions to an access control mechanism [5]. However, when data volumes and sharing grows as fast as in online social networks or smart cities, it becomes difficult to manage access control and deal with personal data [2], [9]. A possible approach would be to securely store access control policies on DLTs, whereby the applicant can be made aware of his or her permissions to access his or her personal data, as in [10], [11]. In particular, Yan et al. [11] employ the use of a SS scheme to share personal information in pieces between network nodes, however their innovative solution is expensive and not GDPR compliant due to the storing of personal data on the DLT.

Another possible approach is to program access control policies as smart contracts, in order to manage control automatically [1], and work on the breaking points between GDPR and DLTs. For this matter, Onik et al. [12], propose a model that stores personal data off-chain (i.e. not directly stored in the DLT) and traces its life cycle through data processors and processors by means of a DLT.

IV. Decentralized Authorization Service

A. Data Storage System

DLTs got momentum, mainly for their ability to provide immutability. However, if we intend to comply with the GDPR, it requires the modification or deletion of data under certain circumstances, e.g. the "right to be forgotten". Thus, in our system, personal data are stored *encrypted* in an off-chain File Storage (FS) and then referenced in a DLT [13]. This solution has the additional benefit of improving performances and to provide higher availability for data reads and writes [2], and can be implemented either by using a DFS or a commercial cloud storage provider, since data is encrypted in the client device. Once a file is published in the off-chain storage (1st step in Figure 1), the returned reference can be employed to retrieve it and its digest allows the verification of its integrity [4]. We implement pepper and salt in the hash function and when possible we try to consider a sufficiently large parameter space of the original data in order 'not to allow the data subject to be identified via "all" "likely" and "reasonable" means' [13], [14].

B. Smart Contract Access Control

Ethereum is the DLT where part of out the access control logic to share data is performed. Through smart contracts, access to the data can be purchased or can be allowed by the owner (2nd step in Figure 1). The use of data, then, is authorized only to entitled users. Hence, due to the presence of smart contracts, no direct interactions are needed among the data owner and users interested in his data. In practice, each piece of data is referenced in a specific smart contract in Ethereum. The smart contract maintains an Access Control List (ACL) that represents the rights to access a bundle of data. Once a consumer is eligible to obtain certain data, i.e. he is in the ACL, he can access such data through an access key, which is provided by the authorization service.

C. Cryptosystem

We use a hybrid encryption scheme [15] that consists of an asymmetric public key part to encrypt a key, using a Key Encapsulation Mechanism (KEM), plus a symmetric secret key part to encrypt actual data through a Data Encapsulation Mechanism (DEM). This KEM/DEM technique leverages the combination of the efficiency and large message space of secret key cryptography with the benefits of public key cryptography [15]. According to this approach, data generated from a data source device is encrypted using a symmetric

[4]The full description of this data storage system is available in [2], [3] since the main focus, here, is on the authorization service.

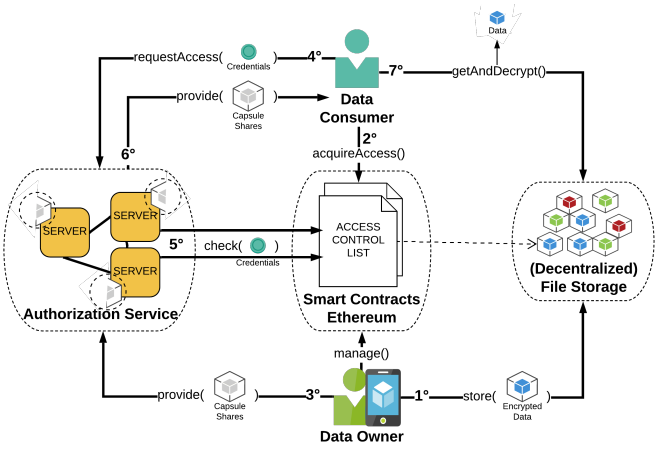

Fig. 1. Architecture of the Decentralized Authorization Service

content key k_{DEM} in an efficient DEM. This key is stored in a "capsule" generated by a KEM, i.e. it is encrypted using public key cryptography with a keypair (pk_{KEM}, sk_{KEM}). In order to efficiently cope with time series data, we refer to Droplet's key management [16] to handle symmetric content keys in the DEM and on their use of Dual-Key Stealth Address Protocol to protect the privacy of sharing relationships in public DLTs.

D. Authorization Service Network

In our proposal, the authorization service is in charge of enforcing the access rights that are specified in the smart contracts ACLs. When this service is operated by a single central provider, trust must be given to this one, since the keys are kept in one place only. Assuming that this provider can be honest-but-curious, privacy may be threatened, e.g. an online social network site sharing a user geolocation with his/her friends, if curious, can access to this information. Thus, we propose to decentralize the service in order to shift the trust to the protocol. In this case, nodes in a network are considered semi- or un-trusted, but a data protection/cryptographical mechanism, built into their execution protocol, allows the whole system to be trusted [6], [7], [10], [17].

When a data consumer with keypair (pk_c,sk_c) is entitled to access some data in a smart contract ACL, he requests the release of the associated capsule to the authorization service through a message signed with pk_c, that proves that his address is in the ACL (4$^{\text{th}}$ step in Figure 1). Upon user request, the authorization service checks if the data consumer is eligible, through interaction with the smart contract (5$^{\text{th}}$ step). If this is the case, i.e. the data consumer is on the ACL, the service starts the operation for releasing, or "opening", the capsule (6$^{\text{th}}$ step) that holds the k_{DEM} secret key needed to decrypt the desired data publicly stored in a (D)FS (7$^{\text{th}}$ step). We refer to two cryptographical schemes for the data owner's key management (3$^{\text{rd}}$ and 6$^{\text{th}}$ step):

1) Secret Sharing (SS): This scheme splits the sk_{KEM} in n shares, but only t shares ($t < n$) are enough to "open" the capsule, i.e. decrypting the capsule in order to obtain k_{DEM}. A (t, n)-threshold scheme is employed, thus, single nodes alone are unable to reconstruct sk_{KEM}, because they only save a portion of this key.

2) Threshold Proxy Re-Encryption (PRE): The capsule, initially obtained from the pk_{KEM} by the data owner, can be re-encrypted by the service using a re-encryption key $pk_{O\to C}$ generated by the owner. The re-encrypted capsule, then, can be decrypted using sk_c by the consumer to obtain the k_{DEM} needed to decrypt the data. PRE usually involves only one semi-trusted proxy node, however, it can decide to not follow the conditional policies as instructed or it may collude with the consumer to attack the data owner's private key. Instead of using a single re-encryption key, multiple proxies can be involved in a (t, n)-threshold scheme with "re-encryption shares", in such a way that these can be combined client-side by the data consumer.

Both these techniques come with different advantages and disadvantages. SS relieves the user from any interaction during each key distribution, but at the same time if t nodes are malicious then the user cannot intervene to stop the keys from getting leaked. On the other side, PRE has the drawback of requiring the user to generate a re-encryption key $pk_{O\to C}$ for each new consumer, however he has the option to stop producing new re-keys if some nodes are malicious.

V. Performance Evaluation

We measured the amount of time required to perform access control operations using an implementation of the proposed systems. We resort to the SS feature provided by the OpenEthereum client [18] and to the PRE implementation of NuCypher [17]. The tests were performed using a network of 25 interconnected nodes with the aim to emulate the real DLTs and the distributed systems use cases [5]

We considered the use case where a data owner stores a smart contract containing an ACL in the Ethereum blockchain. Then, we emulated from 10 to 100 data consumers asking for access to some data after they have been added to the ACL. The results of the test carried out allow us to evaluate the goodness of the proposed approach in terms of performances:

- **Threshold variation**: involves the variation of t from 1 to 25, with number of nodes $n = 25$ and message size set to 30 Bytes. As the first plot in Figure 2 shows, the encryption time remains mostly constant (∼7ms for PRE and ∼52ms for SS) while the decryption time increases linearly with t. The biggest time difference comes from the actual generation and distribution of the key shares in the encryption, ∼792ms, in favor of PRE.
- **Number of nodes variation**: threshold value t was set to 2 and the message size was set to 30 KB. Generally, as expected, the time costs of operations increase with the number of nodes n. However, we must note the fact that the slope of the curve in the second plot in Figure 2 representing SS results is significantly higher than with PRE. This makes the PRE method more scalable.

[5]The reference software for tests can be found in Zenodo https://doi.org/10.5281/zenodo.4024735

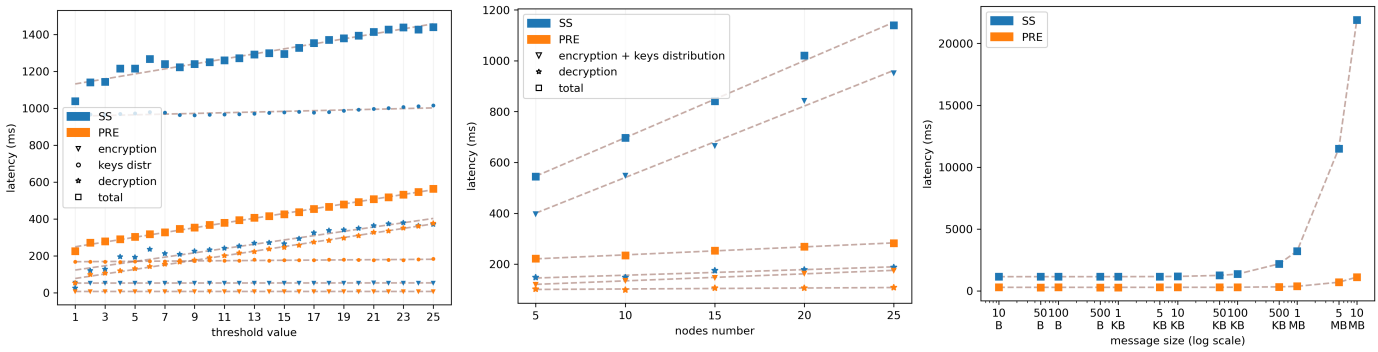

Fig. 2. Encryption and decryption latencies while varying the threshold value, nodes number and messages size.

- **Size of messages variation**: n was set to 25 and $t = 2$, while the size of the message varied. Results reported in the third plot Figure 2 suggest that the PRE scheme scales beter than SS. From 10 Bytes to 1 MB PRE latency raises slightly, while SS has a clear inflection point when the message size is set to 100 KB and then skyrockets from 1 MB onward.

VI. Discussion and Conclusion

In this paper, we presented the architecture of a PIMS, based on a decentralized approach for managing access to data where several entities have the role of providing mutual trust between the parties, i.e. the Authorization Service Network. We leveraged smart contracts to allow individuals to define access through an ACL to their personal data stored off-chain.

We have focused on data protection through encryption, using two different schemes: SS and PRE. We showed their employment in the PIMS architecture and at first we discussed their qualitative differences, then we compared them in terms of execution time. Our performance evaluation shows that, in respect to SS, PRE is: (i) faster when increasing the size of the messages; (ii) more scalable, as it better manages the increase in the number of nodes executing the protocol; (iii) more efficient when increasing the threshold value, due to its shares generation method. On the other hand, PRE has the drawback of requiring the data owner to generate a re-encryption key for each new data consumer. We also acknowledge the lack of scalability of the Ethereum blockchain and the expensiveness, in terms of cost per operation, needed to maintain a smart contract.

In future work, we will pursue a more complex policy enforcement, adding another technological layer on top of the solution we presented, and we will compare it with more sophisticated methods, such as Attribute Based Encryption (ABE).

References

[1] M. Davari and E. Bertino, "Access control model extensions to support data privacy protection based on gdpr," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 4017–4024.

[2] M. Zichichi, S. Ferretti, and G. D'Angelo, "On the efficiency of decentralized file storage for personal information management systems," in *Proc. of the 2nd International Workshop on Social (Media) Sensing, co-located with 25th IEEE Symposium on Computers and Communications 2020 (ISCC2020)*. IEEE, 2020, pp. 1–6.

[3] ——, "A framework based on distributed ledger technologies for data management and services in intelligent transportation systems," *IEEE Access*, pp. 100 384–100 402, 2020.

[4] V. Buterin *et al.*, "Ethereum white paper," 2013. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[5] M. Jemel and A. Serhrouchni, "Decentralized access control mechanism with temporal dimension based on blockchain," in *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*. IEEE, 2017, pp. 177–182.

[6] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[7] G. R. Blakley, "Safeguarding cryptographic keys," in *1979 International Workshop on Managing Requirements Knowledge (MARK)*. IEEE, 1979, pp. 313–318.

[8] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Transactions on Information and System Security (TISSEC)*, vol. 9, no. 1, pp. 1–30, 2006.

[9] M. Zichichi, S. Ferretti, and G. D'Angelo, "A distributed ledger based infrastructure for smart transportation system and social good," in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–6.

[10] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *2015 IEEE Security and Privacy Workshops*. IEEE, 2015, pp. 180–184.

[11] Z. Yan, G. Gan, and K. Riad, "Bc-pds: protecting privacy and self-sovereignty through blockchains for openpds," in *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)*. IEEE, 2017, pp. 138–144.

[12] M. M. H. Onik, C.-S. Kim, N.-Y. Lee, and J. Yang, "Privacy-aware blockchain for personal data sharing and tracking," *Open Computer Science*, vol. 9, no. 1, pp. 80–91, 2019.

[13] M. Finck and F. Pallas, "They who must not be identified—distinguishing personal from non-personal data under the GDPR," *International Data Privacy Law*, vol. 10, no. 1, pp. 11–36, 03 2020. [Online]. Available: https://doi.org/10.1093/idpl/ipz026

[14] Agencia Espanola Proteccion Datos, "Introduction to the Hash Function as a Personal Data Pseudonymisation Technique," Tech. Rep., 2019. [Online]. Available: https://edps.europa.eu/sites/edp/files/publication/19-10-30_aepd-edps_paper_hash_final_en.pdf

[15] J. Herranz, D. Hofheinz, and E. Kiltz, "Kem/dem: Necessary and sufficient conditions for secure hybrid encryption," *IACR Cryptology ePrint Archive*, 2006.

[16] H. Shafagh, L. Burkhalter, S. Duquennoy, A. Hithnawi, and S. Ratnasamy, "Droplet: Decentralized authorization for iot data streams," *arXiv preprint arXiv:1806.02057*, 2018.

[17] M. Egorov, M. Wilkison, and D. Nuñez, "Nucypher kms: decentralized key management system," *arXiv preprint arXiv:1707.06140*, 2017.

[18] "OperEthereum Secret Store," July 2020. [Online]. Available: https://openethereum.github.io/wiki/Secret-Store