ID-based user-centric data usage auditing scheme for distributed environments

Nesrine Kaaniche, Maryline Laurent, Claire Levallois-Barth

frontiers in Blockchain

ORIGINAL RESEARCH
published: 28 April 2020
doi: 10.3389/fbloc.2020.00017

ID-Based User-Centric Data Usage Auditing Scheme for Distributed Environments

Nesrine Kaaniche[1], *Maryline Laurent*[2,3]* *and Claire Levallois-Barth*[3,4]

[1] Department of Computer Science, University of Sheffield, Sheffield, United Kingdom, [2] SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, Paris, France, [3] Member of the Chair Values and Policies of Personal Information, Paris, France, [4] Télécom Paris, Institut Polytechnique de Paris, Paris, France

Recent years have witnessed the trend of increasingly relying on remote and distributed infrastructures, mainly owned and managed by third parties. This increased the number of reported incidents of security breaches compromising users' personal data, where involved entities may massively collect and process massive amounts of such data. Toward these challenges, this paper combines hierarchical Identity Based Cryptographic (IBC) mechanisms with emerging blockchain technologies and introduces a blockchain-based data usage auditing architecture ensuring availability and accountability in a personal data-preserving fashion. The proposed approach relies on smart auditable contracts deployed in blockchain infrastructures. Thus, it offers transparent and controlled data access, sharing and processing, so that unauthorized entities cannot process data without data subjects' consent. Moreover, thanks to the usage of hierarchical ID-based encryption and signature schemes, the proposed solution protects and ensures the confidentiality of users' personal data shared with multiple data controllers and processors. It also provides auditing capacities with tamper-proof evidences for data usage compliance, supported by the intrinsic properties of the blockchain technology.

Keywords: blockchain, personal data protection, data usage auditing, hierarchical ID-based Cryptography, user-centric, GDPR, accountability

OPEN ACCESS

Edited by:
Oskar Josef Gstrein,
University of Groningen, Netherlands

Reviewed by:
Aneta Poniszewska-Maranda,
Lodz University of Technology, Poland
Jianwei Liu,
Beihang University, China
Qi Xia,
University of Electronic Science and Technology of China, China

***Correspondence:**
Maryline Laurent
maryline.laurent@telecom-sudparis.eu

Specialty section:
This article was submitted to Blockchain for Good, a section of the journal Frontiers in Blockchain

Received: *25 September 2019*
Accepted: *31 March 2020*
Published: *28 April 2020*

Citation:
Kaaniche N, Laurent M and Levallois-Barth C (2020) ID-Based User-Centric Data Usage Auditing Scheme for Distributed Environments. Front. Blockchain 3:17. doi: 10.3389/fbloc.2020.00017

1. INTRODUCTION

Nowadays, organizations are collecting large amounts of personal and sensitive data about individuals. The most sensitive data are the most valuable for emerging technologies and applications, namely Artificial Intelligence (AI). Medical records, financial statements, location history, or voice transcripts may all be processed by AI algorithms to provide services that improve individuals' daily lives. This raises the question of the transparency of usage and protection of the collected personal data. Indeed, in several settings, users have little or no control over the data collected and stored about them and how they are used.

Several approaches have been introduced in order to address personal data confidentiality issues, from both legislative and technical perspectives. Indeed, strong authentication and authorization mechanisms based on centralized trusted authorities, emerged for protecting the rights and freedom of the citizens especially for ensuring the right of the protection of personal data and privacy.

In 2018, the General Data Protection Regulation (GDPR) came into force for effectively ensuring the protection of the data subject's personal data [Regulation (EU), 2016]. In particular, the regulation clarifies the conditions under which it is compulsory to obtain the consent of the data subject before processing his personal data, especially for sensitive personal data and data relating to minors. The GDPR also introduces the new obligation of accountability for organizations (i.e., data processors and data controllers). Indeed, each entity processing personal data must be able to demonstrate at any times that it is complying with the obligations laid down by the GDPR.

Recently, various accountable technical systems gained an expanding interest, such as *Bitcoin* that permits users to transfer crypto-currencies (i.e., bitcoins) securely without relying on any centralized entities, thanks to a publicly verifiable open ledger, known as *blockchain*. Thanks to their main intrinsic properties, i.e., tamper-proof infrastructure and availability, blockchain technologies are nowadays widely adopted for data accounting and auditing features.

1.1. Contributions

This paper introduces "a blockchain-based scheme for data usage auditing while preserving personal data confidentiality and ensuring continuous data availability. The proposed scheme relies on hierarchical ID-based cryptographic techniques, where a central master authority delegates the process of public/private keys' generation to the different participating entities, based on authentic public elements" (Laurent et al., 2018). ID-based Encryption (IBE) and Signature (IBS) schemes enable the data subject to encrypt sensitive data and sign transactions, relying on a unique—yet un-identifiable—identifier, respectively.

The proposed solution is multi-fold. First, relying on a blockchain infrastructure, it provides a trusted and transparent environment that permits service providers to collect tamper-proof evidence of received users' consent before gathering, storing, and/or processing their personal data. Second, the proposed framework improves the transnational consent secrecy. "That is, every data subject acts as a delegated PKG by computing an ID-based pair of keys to encrypt/sign the data that he intends to share with either a data controller or a data processor. As such, the data access is managed by the data subject. Third, by using a per smart contract ID-based key, we provide a flexible and secure sharing approach" (Laurent et al., 2018). In fact, the distribution of the decrypting keys between the data subject and the authorized data controllers and processors, does not leak the personal data of the data subject. Fourth, compared to closely related techniques, the proposed solution ensures acceptable processing overheads at both the data owner and the service provider sides.

1.2. Paper[1] Organization

Section 2 introduces the problem statement. Section 3 reviews the blockchain-based technology and discusses related work. Section 4 introduces hierarchical ID-based cryptographic techniques. Section 5 discusses design requirements and highlights security and functional concerns. Section 6 details our proposed solution. Section 7 gives a security analysis of our approach and section 8 concludes the paper.

[1]This paper is an extended and revised version of our former conference work accepted in Kaaniche and Laurent (2017b). Some excerpts of previous publications of the authors (Kaaniche and Laurent, 2017b,c; Laurent et al., 2018) are quoted in the paper from time to time.

2. PROBLEM STATEMENT

According to the GDPR, the data subject's consent is given for specific purposes that must be compliant to both the data controller and the data processor. In this context, three main roles are defined. The *data subject* who gives his consent to a *data controller* (i.e., organization, enterprize) for the processing of his personal data, with the possibility to forward them to a *data processor* (i.e., organization, enterprize) that may process data on behalf of the data controller. Here data controllers are responsible for (i) specifying to the data subject the purpose of data collection, (ii) obtaining the data subject's consent, and (iii) processing personal data according to the consented purposes, and not beyond. We note that for ease of presentation, the remainder of the paper refers to the data subject as the data owner and to both the data controller as well as the data processor as the service provider.

From a data owner perspective, there is a need for new security mechanisms that support data accountability and provenance auditing. In a nutshell, these solutions have to ensure that personal data were accessed by data controllers and/or forwarded to data processors. Indeed, it is important to conceive a secure and transparent solution that permits data owners to (i) check that data controllers and processors are correctly using their personal data with respect to the consented purposes, (ii) verify whether data were accessed, processed, or forwarded without their consent, and (iii) withdraw their consent. From a data controller or processor perspective, there is a need to design a trusted and transparent accountability solution that enables them to get a proof of the data owner's given consent prior to gathering, accessing, processing, or storing his personal data.

3. BLOCKCHAIN-BASED TECHNOLOGY

This section reviews blockchain technologies (cf. section 3.1) and discusses related work (cf. section 3.2).

3.1. Background

Blockchain has gained an attractive interest, since 2008, with the bankruptcy of *Lehman Brothers* in the United States. The root source of this economic crisis is the centralized payment system that relies on clearinghouses, acting as intermediate entities between sellers and buyers in an opaque fashion, and adding a significant extra cost to any inter-bank transactions.

"Bitcoin[2] appeared as an innovative technology enabling users to directly transfer cryptocurrencies in between with no intermediaries. It is considered as the first decentralized cryptocurrency transfer system. It relies on cryptographic proofs of work, digital signatures, and peer-to-peer networking to provide a distributed ledger containing transactions, and referred to as a *blockchain*. A blockchain is essentially a public ledger

[2]https://bitcoin.org/en/

of transactions or events recorded and stored in chronologically connected blocks (Swan, 2015; Crosby et al., 2016; Kaaniche and Laurent, 2017b)."

Two approaches, referred to as permissionless blockchains, have emerged to implement decentralized public services and applications.

"The first approach relies on the existing Bitcoin-blockchain and builds a new framework on top of it. The main advantage of this approach is that the Bitcoin blockchain already exists and is adopted by many users, which makes it more secure, transparent and resilient. The disadvantage is that blocks are mined every 10 min, and the Bitcoin scripting language is not Turing-complete (Swan, 2015).

The second approach is to build an alternative blockchain with all the desired features, which promises full decentralization, such as Ethereum[3]. Additionally to functions already supported by other public blockchain platforms such as bitcoin, e.g., mining of the digital currencies and transaction management, Ethereum also provides a contract functionality known as *smart contract*.

Transactions submitted to the Ethereum environment are organized into blocks and chained to each other based on a cryptographic hash function, initially relying on a pre-computed genesis block. Once a block is added to the blockchain, it cannot be modified or removed for two reasons: first, a block modification would lead to wrong verification of the chain of hash values, and second, the block modification would require intensive efforts to change every replicate of the blockchain supposed to be hosted on a large number of independent nodes. The verification of transactions and the addition of new blocks to the blockchain relies on the so called mining process. Indeed, *miners* have to solve a cryptographic challenge and winners are rewarded (Wood, 2014)", Laurent et al. (2018), referred to as PoW.

Recently, permissioned blockchains are gaining an expanding interest across multiple industries. "This concept appeared as a promoting solution for business applications of blockchain technology and distributed ledgers, in which participants do not necessarily have full trust on each, yet requiring some means of identification" (Kaaniche and Laurent, 2017b).

"Unlike permissionless blockchains, there exists a central entity that decides and grants the right to individual peers to participate in the read/write operations. The Hyperledger[4] project is a prominent initiative dedicated to bringing blockchain technologies to business. It provides a modular consensus protocol, such as Byzantine Fault Tolerance (BFT) algorithm, that ensures efficient scalability and performance complexities with thousands of transactions per second (Kaaniche and Laurent, 2017b; Vukolic, 2017)." Hyperledger development and support are today ensured by a large consortium of world-leading companies. Among prominent instances of permissionned blockchains, we can mention IBM Hyperledger Fabric[5] and R3 Corda[6].

[3]https://www.ethereum.org/
[4]https://www.hyperledger.org/
[5]https://www.hyperledger.org/projects/fabric
[6]https://www.corda.net/

3.2. Related Work

Several solutions have been proposed aiming at empowering data owners while giving them -complete- control over their collected and processed personal data. These solutions mainly investigate the mechanisms that help setting-up an informed consent between the data owner and data controllers/processors, for the different provided applications/services (Laurent, 2019; Lee et al., 2019; Morel et al., 2019; Rantos et al., 2019).

Morel et al. (2019) proposed a framework for gathering data owners consents in IoT environments. In their design, the authors assume that each device collecting data has to send an attestation about the collected and generated information. This information includes the device location, the data types, and privacy policies. The data owner receives the information via a smartphone and gives consent if accepts to share his data. Rantos et al. (2019) introduced a cloud-based platform, called ADVOCATE, which allows users to easily access their personal data and manage consents. The gathered data, from IoT devices, are searchable and can only accessed by authorized entities. Data controllers and processors should submit a query when they want to access the data generated by a specific equipment. And, the query has to clearly specify the purpose of access, the purpose of processing, the period of storage, etc. Once a query is negotiated and accepted by the data owner, the consent has to be signed by both the data owner and the data controller, and stored for auditing purposes. The proposed solution suggests the deployment of a blockchain-based infrastructure to ensure that no modification will be made on given consents. In fact, the main intrinsic property of the blockchain is its suitability for data auditing purposes. It has attracted interest of the research community due to its shared and fault-tolerance database (Zyskind et al., 2015; Linn and Koo, 2016; Ouaddah et al., 2016; Fu et al., 2017; Liang et al., 2017; Shetty et al., 2017; Kaaniche and Laurent, 2018; Ramachandran and Kantarcioglu, 2018; Zhang et al., 2018).

"Liang et al. (2017) proposed a blockchain-based data provenance architecture for cloud applications. Their construction records operations' history as provenance data which are hashed into Merkle tree nodes. A list of hashes of provenance data forms a Merkle tree which are attached as a blockchain transaction. As such, it is possible to immutably prove the provenance of data exchanges. Although the proposed approach is novel, it does not cover the definition of advanced policies or contracts regulating the usage of collected data" (Kaaniche and Laurent, 2017b).

Zyskind et al. (2015) presented an hybrid solution for personal-data management. The proposed system combines both blockchain, as an access moderator, and off-blockchain, for data storage purposes. Proposal Zyskind et al. (2015) considers clients as unique owners of their personal data, thus making them aware of all the associated data being collected by service providers and how they are used. That is, when a client subscribes to a service provider, one transaction is created defining the set of access policies and another contains the hashes of data stored in an off-chain database. However, the Zyskind et al. (2015) proposal permits to only define simple permit/deny access policies through white/blacklisting.

Based on Zyskind et al. (2015), "Linn et al. propose an application of the data auditing framework for health scenarios (Linn and Koo, 2016). In their construction, the blockchain is also considered as an access moderator to control the access to outsourced shared data" (Laurent et al., 2018). Fu et al. (2017) introduced a privacy-aware blockchain-based auditing system for shared data in cloud applications. In order to mitigate the power abuse of single tracing authorities, Fu et al. (2017) presents "a threshold approach, where at least t entities have to collaborate to recover the identity of a malicious user, ensuring thus the non-frameability of users. Based on a blockchain architecture, the proposed construction enables group users to trace data changes and recover latest correct data blocks when current data are damaged" (Kaaniche and Laurent, 2017b). Later, "Neisse et al. discussed the design requirements of blockchain-based solutions for data provenance tracking (Neisse et al., 2017), namely client-centric, server-centric, and data-centric approaches. The authors also presented an evaluation of their implementation results, in order to give a comprehensive overview of different defined approaches" (Laurent et al., 2018).

Dorri et al. (2017) introduced an architecture for data access control management in IoT environments, i.e., smart home application. The architecture relies on a central–unique–miner, i.e., local home miner, to mine blocks, and implement the access policy. The proposed solution ensures the confidentiality of data through a predefined policy, however the introduction of a central miner raises the risk of a single point of failure.

Shetty et al. (2017) proposed a blockchain-based auditing framework, called BlockCloud. The proposed solution supports "monitoring user activities in real-time using hooks and listeners which are special classes of event listeners so that every user operation on files will be collected and recorded for generating provenance data. Each piece of provenance information is referred to as transactions that are broadcasted to the core of the blockchain network created by a specific set of validating virtual machines" (Tosh et al., 2018). "The provenance auditor validates provenance data by retrieving transactions from the blockchain network by using blockchain receipt which contains block and transaction information" (Shetty et al., 2017).

Later, Ramachandran et al. introduced SmartProvenance (Ramachandran and Kantarcioglu, 2018), a blockchain-based auditing solution which applies Ethereum smart contracts to support the data provenance feature. Indeed, their proposed solution is based on two smart contracts, namely Document Tracker contract and Vote contract. The Document Tracker smart contract is used to keep track of changes to a given document while the Vote contract implements the voting protocol. That is, "every provenance change event has to be approved through a voting process by the vote contract. The data trails are only logged if they are approved by the Vote contract. The Vote contract implements two types of voting: simple majority voting and threshold voting" (Ramachandran and Kantarcioglu, 2018) where all blockchain network's minors are called to participate in the voting process. SmartProvenance implements a JavaScript client, run on the browser of each of the user's machines. The client, acting as an interface between the user and back-end smart contracts, is in charge of the synchronization with the smart contract, including the storage, the retrieval, and the validity checking of the changes.

Zhang et al. (2018) introduced an auditing scheme for cloud storage environments, based on a blockchain infrastructure. The proposed solution aims at counteracting provenance records' forgery, removal, and modification attacks. That is, each provenance record is anchored into a blockchain-transaction, and all the provenance records associated with a data item constitute a record chain. Thus, if one record entry is corrupted, the chain is broken.

Table 1 provides a comparison between the proposed scheme and most-closely related solutions, in terms of functional and security features and incurred computation overheads at the data owners and processors' sides. The Blockchain technology indicates whether the proposed schemes is relying on a permissionned or permission-less blockchain. The mining process indicates the consensus algorithm. The confidentiality, unlinkability, and anonymity properties (detailed in section 5) refer to the secrecy of personal data, unlinkability between different transactions and anonymity of the data owner, respectively. The performances at the data owner and service providers sides indicate the processing overheads.

4. IDENTITY BASED CRYPTOGRAPHY

Introduced by Shamir (1985), Identity Based Cryptography (IBC) enables the derivation of public and private keys with no need for a certification authority. Indeed, each entity should use one of its—unique—identifiers as its public key. Rather than relying on a Public Key Infrastructure (PKI), IBC assigns the private key generation function to a third party, referred to as a Public Key Generator (PKG). Thus at the request of an entity, the PKG computes the private key associated with his public key.

In order to be able to derive an entity's private key $sk_{\mathcal{E}}$, the PKG must first define a set of *ID-based public elements* (IBC–PE). The PKG generates the groups $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ and the pairing function $\hat{e}$ from $\mathbb{G}_1 \times \mathbb{G}_2$ in $\mathbb{G}_T$. $\mathbb{G}_1$ and $\mathbb{G}_2$ are additive subgroups of the group of points of an Elliptic Curve (EC). However, $\mathbb{G}_T$ is a multiplicative subgroup of a finite field. $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ have the same order q. In addition, $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ are generated by P, Q and the generator $g = \hat{e}(P, Q)$, respectively.

The PKG defines the groups and a set of hash functions in accordance to the selected ID-based encryption and signature schemes. That is, the PKG uses a hash function $Hash_{pub}()$ to transform the entity's identity ($ID_{\mathcal{E}}$) into a public key as follows:

$$pk_{\mathcal{E}} = Hash_{pub}(ID_{\mathcal{E}})$$

Generally, the public key of the entity is computed as a hash of one of his identities.

The PKG generates an entity's private key based on a local secret $s_{PKG} \in \mathbb{Z}_q^*$ and a private key generation function keygen(). The private key is computed as:

$$sk_{\mathcal{E}} = \mathsf{keygen}(s_{PKG}, pk_{\mathcal{E}})$$

The groups $\mathbb{G}_1$ and $\mathbb{G}_2$, the pairing $\hat{e}$, the points P, Q, and $Q_{pub} = s_{PKG}.Q$, and the hash functions form the ID-based public

TABLE 1 | Comparison between the proposed solution and most closely related approaches.

Scheme	Security and functional properties					Performances	
	Blockchain technology	Mining process	Confidentially	Unlinkability	Anonymity	Owner	Service provider
Zyskind et al. (2015)	Public BC	–	✓	✗	✓	$n_P n_S \gamma_{SC}$	$n_U n_S \gamma_{SC}$
Neisse et al. (2017)	Ethereum	POS	✓	✗	✓	$n_P n_S \gamma_{SC}$	$n_U n_S \gamma_{SC}$
Dorri et al. (2017)	Private BC	collaborative	✗	✗	✓	–	–
Kaaniche and Laurent (2018)	Ethereum	POS	✓	✓	✓	$\gamma_{SC} + (3\gamma_E + \gamma_S)$	$\gamma_{SC} + N(6\gamma_E + 3\gamma_S)$
Shetty et al. (2017)	Public BC	– Cloud pricing	✓	✗	✓	–	–
Ramachandran and Kantarcioglu (2018)	Ethereum	POS	✗	✗	✓	$2\ n_P n_S \gamma_{SC}$	$n_U n_S \gamma_{SC}$
Zhang et al. (2018)	Ethereum	POS	✓	✗	✓	$n_P n_S(\gamma_{SC} + N(2\gamma_E + 2\gamma_S))$	$n_U n_S(\gamma_{SC} + N(2\gamma_E + \gamma_S))$
Our solution	Consortium BC HyperLedger Fabric	PBFT	✓	✓	✓	$n_P n_S(\gamma_{SC} + N(\gamma_E + \gamma_S))$	$n_U n_S(\gamma_{SC} + N(\gamma_E + \gamma_S))$

✓and ✗ indicate whether the property is supported or not, respectively. –Indicates that the property is not-applicable or non specified. n_P, n_O, and n_S indicate the number of service providers, owners, and services/applications, respectively. γ_{SC}, γ_E, and γ_S indicate the processing overhead of a smart contract creation, data encryption/decryption, and data signature/verification, respectively. N is the number of transactions associated with a smart contract.

elements as follows:

$$IBC - PE = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, \hat{e}, g, P, Q, Q_{pub},$$

$$Hash_{pub}(), H_1(), ..., H_k()\}.$$

ID-based cryptography suffers naturally from the key escrow attack as a PKG knows the owners' private keys and, if untrusted, can realize masquerading attacks. To mitigate the key escrow attack, a hierarchy of PKGs has been proposed in Gentry and Silverberg (2002) and Horwitz and Lynn (2002), along with the Hierarchical Identity Based Cryptography (HIBC) to reduce the workload of the PKG by delegating the private key generation task to lower level entities, i.e., entities who have already obtained their private keys. As such, Hierarchical Identity based Encryption (HIBE) (Horwitz and Lynn, 2002; Gentry and Halevi, 2009; Blazy et al., 2014; Seo and Emura, 2015) and Hierarchical Identity Based Signature (HIBS) (Gentry and Silverberg, 2002; Chow et al., 2004; Tian and Huang, 2014) schemes emerged, as a generalization of IBE and IBS, respectively, to mirror an organizational hierarchy. That is, an identity at level k of the hierarchy tree can issue private keys to its descendant identities, but it cannot decrypt messages intended for other identities.

HIBE schemes rely on four randomized algorithms, namely: `setup`, `keygen`, `encrypt`, and `decrypt`. The `setup` algorithm generates system parameters, denoted by IBC–PE and the master secret key s_{PKG}. Note that the master key corresponds to the private key at depth 0 and note that an IBE system is a HIBE where all identities are at depth 1. The `keygen` algorithm takes as input an identity $ID = \{ID_1, \cdots, ID_k\}$ of depth k and the secret key $sk_{\mathcal{E}|k-1}$ of the parent entity $\{ID_1, \cdots, ID_{k-1}\}$ of depth $k-1$. It outputs the secret key $sk_{\mathcal{E}|k}$. Similarly to IBE, the encrypt algorithm encrypts messages for an identity using IBC-PE and the decrypt algorithm decrypts ciphertexts using the private key.

5. TECHNICAL DESIGN REQUIREMENTS

In this section, we first review different security and functional requirements (cf. section 5.1). Then, we discuss blockchain design directions (cf. section 5.2) and deployment models (cf. section 5.3).

5.1. Security and Functional Requirements

Our blockchain-based data usage auditing solution has to consider a set of security and functional properties, defined as follows:

- confidentiality—the proposed approach has to "prevent unauthorized disclosure of both personal identifying information and shared data between data owners and service providers" (Kaaniche and Laurent, 2017b).
- unlinkability—the proposed solution has to guarantee that created smart contracts by the same data owner can not be linked by public verifiers as well as service providers.
- auditability—the proposed solution has "to enable auditing authorities to lead an investigation and obtain consistent proofs in case of non-compliant activities" (Kaaniche and Laurent, 2017b). The auditing process, carried by authorized authorities, may be both public and private.
- censorship resistance—this property defines the ability to prevent a third-party from pushing false information across the system (Perng et al., 2005). In fact, service providers are interested by data owners' contract creation and joining, thus allowing them to manage and process data. Otherwise, data owners have the right to revoke or modify their initial consent

agreement. As such, service providers will no longer have access to data, with respect to their granted privileges.
- data transparency—each data owner should have a complete transparent view over how data are collected, accessed, and processed.
- computation overhead—the proposed solution should offer acceptable computation costs.

5.2. Blockchain Design Directions and Discussions

"To efficiently propose the contract design for each participating entity, we have to take into consideration (i) the contract lifecycle, (ii) accountability granularity, (iii) required information to be stored in the contract, (iv) access patterns w.r.t. read/modify granted privileges, and (v) blockchain architecture w.r.t. different models of deployments" (Kaaniche and Laurent, 2017b).

For this purpose, as discussed in Kaaniche and Laurent (2017c), we distinguish "three different cases for designing contracts that depend on the number of involved data owners and processors, as follows:

- M_1—*one data owner contract per specific data:* each data owner has to create a contract for each data instance for all data controllers, while specifying access privileges. In a nutshell, the contract should contain the list of authorized data controllers as well as their respective granted privileges.
- M_2—*one data owner contract for each specific data controller/processor:* the data owner creates a contract for each specific controller processing his personal data. Thus, the contract includes shared data, access policy w.r.t. data usage as well as data usage logs that represent the different events carried out by the data controller on the owner's personal data. This contract design model holds the largest number of contracts and is merely adequate for high security levels needed, for instance, for e-health data.
- M_3—*one data controller contract for multiple data owners w.r.t. data usage policy:* for this sub-case, the data controller creates a generic contract that specifies the manner of processing data received from data owners. When a data owner joins the contract established by the data controller, he then accepts the policy usage identified by the controller. This model is of interest when expecting a lower number of different owners."

"Let us note that the first model (M_1) cannot ensure the unlinkability property because the data owner is known to all organizations under the same identity. Other cases (i.e., M_2 and M_3) allow data owners to protect their identities from linkability attacks, thanks to the use of different identifiers (i.e., blockchain addresses). For instance, the first contract design model (M_1) is more scalable, compared to other design models (M_2) and (M_3), thanks to the use of a generic instance for data.

It is worth noticing that the aforementioned models are different in terms of contracts' number and data auditing granularity allowed to data owners, depending on the required security properties" (Kaaniche and Laurent, 2017c).

5.3. Public vs. Semi-public vs. Private Blockchain

As stated above and as discussed in Kaaniche and Laurent (2017c), "it is important to consider the blockchain architecture (public, semi-public, private), as it impacts several security requirements, mainly public verifiability, unlinkability, and censorship resistance properties (Swan, 2015).

For instance, a public blockchain architecture, such as Bitcoin and Ethereum, allows every entity to read, send transactions and participate in the mining process to decide which blocks are added to the chain and determine the current blockchain state. As such, these public distributed ledgers permit to have a fully transparent system, but with a questionable level of data owners' linkability. In addition, transactions' fees have to be paid for the processing of contract invocations, which could make some approaches unfeasible due to the high number of transactions, like for the data owner-centric models (M_1, M_2). This would make necessary that data owners and service providers agree on an adequate business model to avoid owners supporting all the transaction' fees.

In a private blockchain architecture, read access may be public, or restricted, while write privileges have to be granted by a central entity. In fact, the central entity is responsible for verifying transactions' compliance and censorship resistance with respect to data owners' policies. Hence, a private system provides limited transparency, but with significant unlinkability and personal data protection guarantees. Nevertheless, in a private blockchain, processing fees may be significantly reduced, mainly when considering the service provider-centric model M_3.

In a semi-public or consortium blockchain infrastructure, read access may be public or restricted and the consensus process is controlled by a set of organizations. Thus, a consortium blockchain architecture inherits several benefits from a private blockchain technology—efficiency, unlinkability, and personal data protection guarantees—without relying on one single deciding authority. Additionally, these permissioned blockchains often expose some low-level trust assumptions originating from their consensus mechanism to their smart contract applications, hence ensuring transparency and auditability. Apart from these advantages, a consortium blockchain permits to get reduced transactions' fees thanks to the consensus algorithm, which could be considered as a motivating argument for data-owner centric models, namely M_2."

6. A NEW BLOCKCHAIN-BASED DATA USAGE AUDITING ARCHITECTURE

Our architecture considers that one smart contract per data controller/processor is managed by the data owner (M_2 model of section 5.2), and relies on a consortium-blockchain infrastructure, as discussed in section 5.3.

6.1. Overview

Our blockchain-based data usage auditing solution involves the three following entities: a data owner ($\mathcal{O}$), a service provider ($\mathcal{S}$)

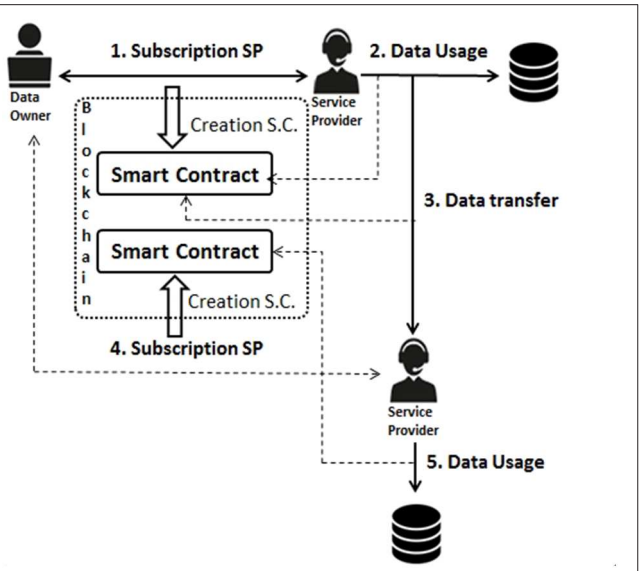

FIGURE 1 | Blockchain-based architecture for data usage auditing schemes.

(acting as a data controller), and a service provider ($\mathcal{P}$) (acting as a data processor) [Regulation (EU), 2016].

Figure 1 depicts the different actors and their interactions. When subscribing to a service provider $\mathcal{S}$, the date owner $\mathcal{O}$ creates first a data usage contract. In this contract, $\mathcal{O}$ provides "the usage policy for both data transferred by $\mathcal{O}$ to $\mathcal{S}$, as well as any other collected data by $\mathcal{S}$ during the interactions with $\mathcal{O}$ (e.g., logging files). When processing $\mathcal{O}$'s data, $\mathcal{S}$ has to comply to the granted usage policy, as registered in the smart contract in the blockchain. In addition, $\mathcal{O}$ needs to identify the list of possible actions that might be performed on behalf of the smart contract, including the smart contract approval (cf. section 6.4).

If authorized by the data usage policy in the smart contract, $\mathcal{C}$ might transmit $\mathcal{O}$'s data to service provider $\mathcal{P}$, by pushing that forwarding information to the blockchain. $\mathcal{O}$ being notified by the forward event, then has to create a new smart contract with $\mathcal{P}$" (Laurent et al., 2018).

As our approach leans on a consortium-blockchain infrastructure, for anyone to be able to read the contract as well as associated transactions, we introduce cryptographic mechanisms—hierarchical ID-based encryption (HIBE) and signature (HIBS) schemes—in order to provide secrecy of exchanged data and preserve data owner's privacy.

Our approach based on hierarchical IBC has several advantages. First, each data owner $\mathcal{O}$, as part of the blockchain, acts as a delegated PKG, thus being enabled to derive an ID-based pair of keys for encrypting and signing all the data intended to be shared with $\mathcal{S}$ or $\mathcal{P}$. It is noteworthy that the data owner $\mathcal{O}$ remains fully in charge of his data access.

Second, using hierarchical identity based schemes introduces the existence of a root PKG entity (i.e., a trusted central entity), which has the responsibility to generate certified public system parameters IBC–PE, for the entities of the system ($\mathcal{O}$, $\mathcal{P}$, $\mathcal{S}$) and to guarantee authentic entities identities within the system. Furthermore, hierarchical identity has the advantage over the client-PKG classical approach—to provide public system parameters common to any entities and not specific to the public identity of the data owner $\mathcal{O}$. "This feature enables $\mathcal{O}$ to get rid of the cumbersome tasks of generating and publishing each $\mathcal{O}$'s own public parameters, and it provides indistinguishability among entities, mostly during the signature verification processes" (Laurent et al., 2018).

Third, having a per smart-contract ID-based key provides our solution with several properties of interest, including indistinguishability among smart contracts at the blockchain level, and unlinkability of smart contracts to data owners. Both properties prevent any reidentification of data owners thanks to search operations over the blockchain.

6.2. Security and Design Assumptions

This section first details the security assumptions in section 6.2.1. Then, it introduces the smart contract design assumptions 6.2.2.

6.2.1. Security Assumptions

Our scheme considers the following security assumptions:

- an off-blockchain secure communication—a secure channel is established between $\mathcal{O}$ and the service provider $\mathcal{P}$ or $\mathcal{S}$, thus enabling $\mathcal{O}$ to securely transmit its personal information along with some data that he expects to share with the service provider. The secure channel supports mutual authentication and data confidentiality and integrity. It can be implemented through the Transport Layer protocol (TLS) (Dierks and Rescorla, 2008), where the data owner can authenticate with a certificate or password. If higher data preservation is requested, then attribute based credential mechanisms, such as Idemix[7] or UProve[8], are good candidates for enabling mutual authentication while revealing only required information to any service provider.
- a robust blockchain and smart contract implementation—any blockchain related operations, including mining and transaction anchoring activities are assumed to be secure and incorruptible, and blockchain to deliver non-tamper proofs of data processing and managing events.
- a trusted PKG entity—the PKG plays a central role, by enrolling the different entities ($\mathcal{O}, \mathcal{S}, \mathcal{P}$) into the system, by deriving their private keys, and by issuing and publishing the security ID-based public elements for the whole ID-based system. In the sequel, in order to strengthen security guarantees and prevent attacks against this central entity, the root PKG functions w.r.t. the derivation of private keys may be distributed among several entities, based on secure multi-party computation mechanisms or threshold techniques (Yu et al., 2010; Prabhakaran and Sahai, 2013; Grumăzescu et al., 2015). Note that, in practice, entities should agree on the hierarchical ID-based encryption and signature schemes which will be used for ciphering and signing messages, respectively. Our proposal does not rely on a specific scheme, however, that choice has a direct impact on the method for generating private keys.

[7]https://www.zurich.ibm.com/identity_mixer/
[8]https://www.microsoft.com/en-us/research/project/u-prove/

6.2.2. Smart Contract Design Assumptions

"For our solution, each data contract involves two parts: *data structure*, referred to as $\mathtt{p}_1$ and *data usage policy*, denoted by $\mathtt{p}_2$. The *data structure* part, includes types and instances of exchanged data between the data owner $\mathcal{O}$ and the service provider $\mathcal{S}$" (Laurent et al., 2018). That is, the data type field contains a list of all data instantiations (i.e., string, integer, date, ...), while data instances provide data values. The *data usage policy* part identifies all the authorized and denied usage actions for both transferred and implicitly collected data. Among data usage actions, we may mention data storage duration granted to the service provider, type of processing permitted over the owner's data, permission to forward data to other service providers or to generate derived data from some exchanged personal data [Regulation (EU), 2016].

Furthermore, for each created smart contract, the data owner $\mathcal{O}$ is required to list the actions that are authorized by the smart contract from $\mathcal{O}$ and service providers ($\mathcal{S}$ and $\mathcal{P}$). Permitted actions from $\mathcal{O}$ include deletion of the smart contract, inactivation of the contract and modification of new data usage policies, and permitted actions from service providers include only approval of the smart contract content.

6.3. System Initialization

We assume that a trusted root PKG entity is in charge of generating and publishing the ID-based public elements, along with the ID-based Signature and Encryption scheme (cf. section 6.2.1), as follows:

$$IBC - PE = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, q, \hat{e}, g, P, Q, Q_{pub}, Hash_{pub}(), H_1(), ..., H_k(), IBS_{scheme}, IBE_{scheme}\}$$

Moreover, each entity $\mathcal{E}$ owns a pair of public and private keys $(sk_{\mathcal{E}}, pk_{\mathcal{E}})$, where $\mathcal{E} \in \{\mathcal{O}, \mathcal{S}, \mathcal{P}\}$. Each pair of public and private keys $(sk_{\mathcal{E}}, pk_{\mathcal{E}})$ is mathematically generated by the root PKG in accordance with the selected ID-based cryptographic schemes, as detailed in section 4. Then the root PKG securely transmits the private key $sk_{\mathcal{E}}$ through the secure channel (cf. section 6.2) to the entity $\mathcal{E}$. As such, $\mathcal{E}$ owns his secret $sk_{\mathcal{E}}$ associated to his public key $pk_{\mathcal{E}}$ where $sk_{\mathcal{E}}$ is derived from the unique entity identity of $\mathcal{E}$ (i.e., one of $\mathcal{E}$'s blockchain addresses), based on the PKG's own secret and the public elements IBC–PE. "As such, the pair $(pk_{\mathcal{E}}, sk_{\mathcal{E}})$ is somewhat certified by the root PKG, as the signature generation and data decryption operations require an entity to be equipped with his own secret $sk_{\mathcal{E}}$" (Laurent et al., 2018).

6.4. Smart Contract Creation

Figure 2 gives the full picture of the smart contract creation from the data usage request by $\mathcal{S}$ to $\mathcal{O}$ up-to the approval of the contract by $\mathcal{S}$ and transmission of data to $\mathcal{S}$.

First, $\mathcal{O}$ establishes a direct secure channel with $\mathcal{S}$ where $\mathcal{S}$ and $\mathcal{O}$ authenticate each other thanks to their respective ($sk_{\mathcal{E}}$ and $pk_{\mathcal{E}}$) pair of keys (cf. section 6.3). The transaction is identified with an T_{id} parameter. Through this channel, $\mathcal{S}$ might initiate a data usage request to $\mathcal{O}$, for $\mathcal{O}$ to infer some parameters for the smart contract $\mathcal{C}$ prior to launching the smart contract into the blockchain. The generation of these parameters is presented in section 6.4.1, and details of smart contract creation are given in section 6.4.2.

6.4.1. Generation of Per Smart-Contract Parameters by $\mathcal{O}$

After the initialization phase, as described in section 6.3, $\mathcal{O}$ owns his public and private keys $(sk_{\mathcal{O}}, pk_{\mathcal{O}})$, thus making $\mathcal{O}$ able to derive a per smart-contract $ID_{\mathcal{C}}$, with its associated pair of public and private keys $(sk_{\mathcal{C}}, pk_{\mathcal{C}})$, as follows.

Based on the public elements IBC–PE, and the hierarchical ID-based principle, as detailed in section 4, $\mathcal{O}$ first generates an identifier $ID_{\mathcal{C}}$, that constitutes the hierarchical smart contract identity of depth 2 which relies on both $\mathcal{O}$'s identity $ID_{\mathcal{O}}$ and the smart contract identifier $ID_{\mathcal{C}}$.

Then, thanks to `keygen`, $\mathcal{O}$ derives a per smart contract ID-based public key $pk_{\mathcal{C}}$ based on the general public elements IBC–PE and the concatenation of the data owner's address $ID_{\mathcal{O}}$, the service provider address $ID_{\mathcal{S}}$ and the smart contract identifier $ID_{\mathcal{C}}$.

$$pk_{\mathcal{C}} = Hash_{pub}(ID_{\mathcal{S}}||ID_{\mathcal{C}})$$

Note that the secret key $sk_{\mathcal{C}}$ associated with the public key $pk_{\mathcal{C}}$ is only known by the data owner, at the moment of the smart contract creation.

6.4.2. Creation of the Smart Contract by $\mathcal{O}$

$\mathcal{O}$ approves first the data usage request and then constructs the data section part of the contract by distinguishing parts $\mathtt{p}_1$ and $\mathtt{p}_2$, as presented in section 6.2.2. To avoid publishing personal information, our approach proposes to obfuscate data values in $\mathtt{p}_1$ thanks to the use of the public hash function $\mathcal{H}$ (e.g., SHA-256). To counteract the linkability attacks, we propose to concatenate data values with a symmetric key $k_{\mathcal{C},\mathcal{S}}$ derived from the generated smart contract secret key $sk_{\mathcal{C}}$ before application of the hash function. The derivation of $k_{\mathcal{C},\mathcal{S}}$ has to be unidirectional, through a hashing function for instance. "Then, $\mathcal{O}$ generates a smart contract creation transaction, which will be anchored in the blockchain. As such, we define a smart contract creation transaction T as the tuple $T = [T_{id}, ID_{\mathcal{C}}, U_{\mathcal{C}}, \mathtt{p}_1, \mathtt{p}_2, enc_{pk_{\mathcal{S}}}(k_{\mathcal{C},\mathcal{S}}), \sigma_{\mathcal{C}}]$, where T_{id} is the transaction identifier, $ID_{\mathcal{C}}$ is the smart contract identifier, $U_{\mathcal{C}}$ is a smart contract approval request identifier toward $\mathcal{S}$, $enc_{pk_{\mathcal{S}}}(k_{\mathcal{C},\mathcal{S}})$ is the encrypted symmetric key $k_{\mathcal{C},\mathcal{S}}$, based on the public key of the service provider $\mathcal{S}$ and $\sigma_{\mathcal{C}}$ is the signature of $\mathcal{O}$ of all the previous fields, using the private key $sk_{\mathcal{C}}$, only known to the data owner" (Laurent et al., 2018).

After being notified of the smart contract creation request, the service provider decrypts the enciphered (pairwise) symmetric key $k_{\mathcal{C},\mathcal{S}}$ using the public system parameters IBC–PE, and his own private key $sk_{\mathcal{S}}$. Then, $\mathcal{S}$ verifies the signature of the issuing data owner using the public key $pk_{\mathcal{C}}$. Only if both operations are successful, $\mathcal{S}$ can approve the transaction associated with the smart contract creation thanks to its own key $sk_{\mathcal{S}}$.

Upon receiving the smart contract approval notification, $\mathcal{O}$ transmits his personal data to the service provider, through a secure channel. The compliance between his personal data and the hashed values embedded in $\mathtt{p}_1$ proves his consent agreement.

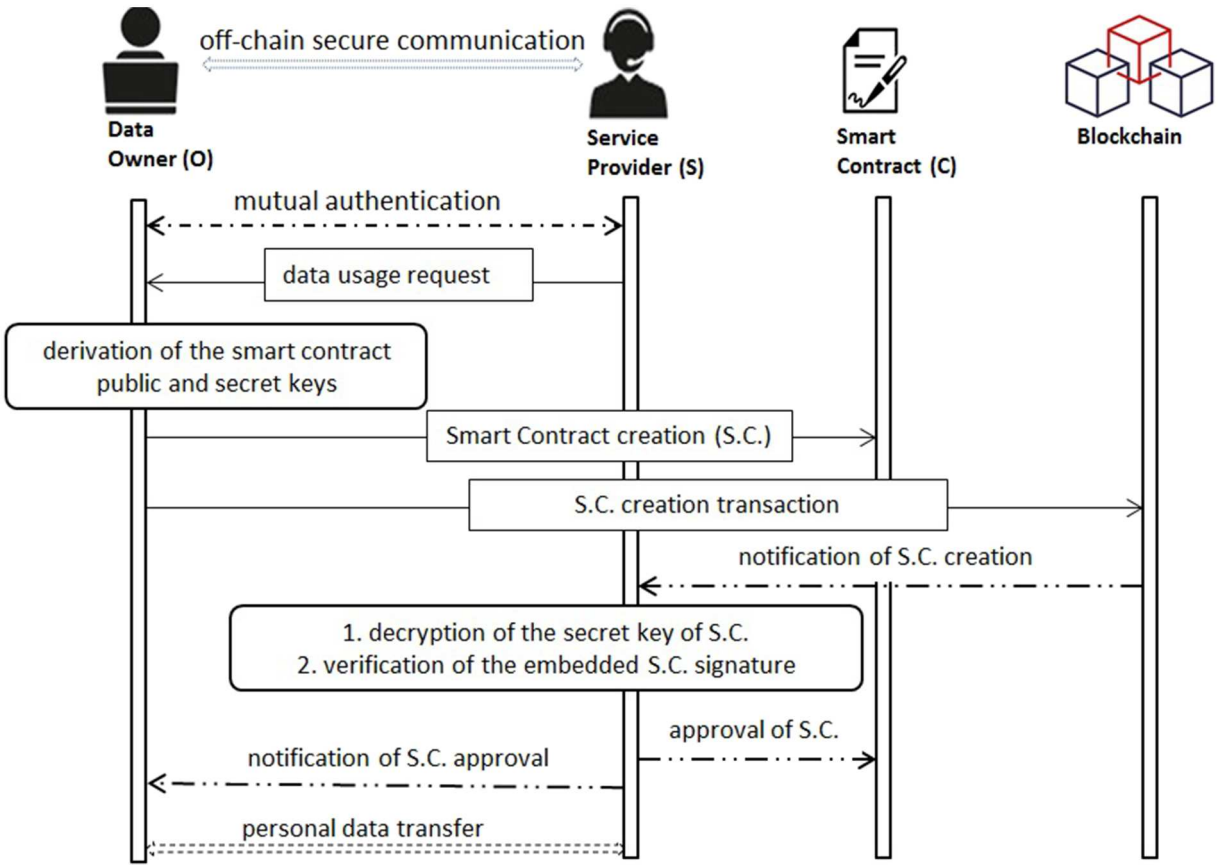

FIGURE 2 | Smart contract creation process.

Then, $\mathcal{S}$ can check the integrity of the received $\mathcal{O}$'s data based on the deciphered pairwise key $k_{\mathcal{C},\mathcal{S}}$ and the received data values.

6.5. Data Usage Transfer

"In case a forward action is authorized on some $\mathcal{O}$'s data by the data usage policy specified in the smart contract with $\mathcal{S}$, then data might be transferred by the service provider $\mathcal{S}$ to the corresponding data processor $\mathcal{P}$ and a transaction has to be pushed to the blockchain" (Laurent et al., 2018).

"For this purpose, $\mathcal{S}$ has to generate a data usage transaction. As such, the data owner is notified about the service provider data transfer activity. We define a data usage transaction T corresponding to a data transfer activity as the tuple $T = [T_{id}, ID_{\mathcal{C}}, ID_{\mathcal{S}}, U_{\mathcal{C}}, enc_{pk_{\mathcal{O}}}(ID_{\mathcal{P}}), \sigma_{\mathcal{S}}]$, where T_{id} is the transaction identifier, $ID_{\mathcal{C}}$ is the smart contract identifier, $ID_{\mathcal{S}}$ is the service provider identity, $U_{\mathcal{C}}$ is a transfer data event identifier, $enc_{pk_{\mathcal{O}}}(ID_{\mathcal{P}})$ is the encrypted identity of $\mathcal{P}$, based on the public key of the data owner $\mathcal{O}$ and $\sigma_{\mathcal{S}}$ is the signature of $\mathcal{S}$ over all the previous fields" (Laurent et al., 2018).

The data owner $\mathcal{O}$ being notified of the data transfer creates a new smart contract with data processor $\mathcal{P}$, following the procedure presented in section 6.4. Note that $\mathcal{P}$ does not know the identifier of the newly agreed contract between $\mathcal{O}$ and $\mathcal{S}$.

6.6. Other Operations Over the Smart Contract

Additionally to the approval of the smart contract by $\mathcal{S}$ or $\mathcal{P}$, the smart contract has to include the following actions:

- deleting the contract from the blockchain by $\mathcal{O}$—In case $\mathcal{O}$ wants to withdraw his data usage consent, the contract must be made inactive, thus leading to denying any subsequent data usages. The only possible remaining activity is the registration of the complete contract history in the blockchain.
- changing data usage—At any moment, $\mathcal{O}$ is able to restrict or expand his data usages.

Any contract policies modifications must be taken into consideration in subsequent data usages. This is made possible thanks to the blockchain notifying any changes w.r.t. the smart contract, as well as data usage policies embedded into $\mathtt{p}_2$, to the involved entities.

6.7. Auditing

The auditing process can be realized either in a private manner by the data owner $\mathcal{O}$ or by a specialized auditing authority; or publicly by anyone.

Public auditing only counts on the transactions' information being readable in the blockchain. As such, a public verifier can detect some non-compliant activities, by comparing any transactions where a specific data usage activity is reported, with the smart contract it refers to through its identifier. Indeed, in case the smart contract does not include the data usage activity reported in the transaction, a compliance issue is revealed. This auditing service is made reliable thanks to the tamper-proof feature of the blockchain, and any transactions being signed by the service provider.

Private auditing enhanced by a dedicated auditing organization, relies on public blockchain information, as well as private data, provided by the claimer. For instance, when a data owner requests a private audit for a misuse of his personal data, he shares the private and public keys associated to the concerned smart contracts $(pk_{\mathcal{C}_i}, sk_{\mathcal{C}_i})_{\{i\in[1,N]\}}$, where N is the number of audited smart contracts. As such, the auditing authority is able to lead an investigation, while crawling blockchain transactions corresponding to the provided smart contracts' identifiers. Having received the smart contracts' private keys $sk_{\mathcal{C}_i}$ where $i \in [1, N]$, the auditing authority is able to interpret the content of blockchain transactions and to detect non-compliant activities. We note that private auditing has to be paid by the audited service provider, if non-compliant activities are reported.

7. SECURITY ANALYSIS

In this section, we first present our threat model. Then, we discuss the resistance of our construction against data confidentiality, unlinkability, and availability attacks.

7.1. Threat Model

For designing a secure blokchain-based auditing scheme, we consider realistic threat models. For instance, our contribution does not take into consideration malicious services that deviate from the protocols as we emphasize that service providers stress about their reputation, in real-world use cases. As such, we consider that "an attacker is able to read, send and drop a transaction addressed to the blockchain. The attacker targets data owners, service providers as well as the blockchain" (Kaaniche and Laurent, 2017b), as follows:

- based on previous data usage sessions, as well as provided blockchain data, "an attacker tries to impersonate a data owner to afford a honest service provider some rights to be logged into the blockchain without the legal data owner's consent" (Kaaniche and Laurent, 2017b). This attack is considered with respect to the confidentiality and auditability requirement.
- an attacker attempts to thwart the unlinkability feature w.r.t. data owners. That is, he attempts to link smart contract identifiers or a smart contract to a specific owner.
- an attacker attempts "to prevent the publication of a legitimate transaction in the blockchain. For example, in order to prevent data transfer notification to the data owner, an attacker may try a DoS attack against a data usage event, or attempt a flooding attack on the blockchain with invalid data usage information. This attack is considered against the auditability, the availability and the censorship resistance requirements" (Kaaniche and Laurent, 2017b).

7.2. Security Discussion

This section discusses the security properties, with respect to the defined threat models. Indeed, the confidentiality (section 7.2.1), auditability (section 7.2.3), unlinkability (section 7.2.2), and availability (section 7.2.4) are analyzed while considering different adversaries.

7.2.1. Confidentiality

Our proposed solution is resistant against data secrecy attacks for several reasons here-below listed:

- "only hashed data values and enciphered information published in the smart contract—the client is in charge of hashing his personal data for fulfilling p_1 of the smart contract, and enciphering a secret with $pk_{\mathcal{S}}$ addressed to server $\mathcal{S}$" (Laurent et al., 2018).
- the enciphering key $k_{\mathcal{C},\mathcal{S}}$ only known by a pair of owner and service provider—the owner $\mathcal{O}$ is the only entity owning secret $sk_{\mathcal{C}}$. As a delegated PKG entity, $\mathcal{O}$ is the only entity able to issue the pair of public and private keys $(pk_{\mathcal{C}}, sk_{\mathcal{C}})$ for a specific smart contract. The derived key $k_{\mathcal{C},\mathcal{S}}$ being securely transmitted to the involved service provider, is thus only known to the owner and the service provider.
- one per smart-contract enciphering key—the pair of keys $(pk_{\mathcal{C}}, sk_{\mathcal{C}})$ generated by the owner is specific to a smart contract, and as such can not leak any information, in case they are compromised, about other per smart-contract keys.

As the public ledger only registers hashed data values or encrypted information, no significant information can be learnt from the blockchain.

7.2.2. Unlinkability

Beyond confidentiality guarantee, the unlinkability property is ensured in our approach thanks to the following technical features:

- one smart contract per service provider—each smart contract is specific to a service provider and has its own identifier $ID_{\mathcal{C}}$ and secret key $sk_{\mathcal{C}}$. The owner using this per smart-contract secret key for signing the contract creation can not be identified. It is even not possible to link two smart contracts provided with two different $ID_{\mathcal{C}}$ to the same owner.
- unique hashed values within smart contracts—linking smart contracts in between as issued by the same owner is not possible, as smart contracts always concatenate data values with a specific information they own (their pairwise key $k_{\mathcal{C},\mathcal{S}}$) before hashing. As such, a search over the blockchain ledger for the same data values (assumed to match the same owner) is inconclusive.

7.2.3. Auditability

The proposed approach ensures the auditability requirement as follows:

- tamper-proof architecture—as emphasized in section 6.2.1, "all blockchain-specific operations, such as transaction anchoring activities, are considered as secure and non-corruptible, thus ensuring non-tamper proofs of data processing and managing events" (Laurent et al., 2018).
- transparent usage—our approach is based on a consortium blockchain infrastructure, that permits public access (i.e., read privilege) the contract and its associated transactions, to anyone. Thus, it provides a transparent view over how data are collected and accessed.
- signed transactions—"our approach relies on signed transactions. That is, both smart contract creation and data usage transactions have to be signed by the data owner $\mathcal{O}$ and the service provider $\mathcal{S}$ and $\mathcal{P}$, respectively. Signed transactions ensure that each activity has been efficiently performed by the holder of the used private key, which is certified by the PKG entity. As such, the resistance of the chosen HIBS scheme against forgery attacks has a direct impact on the fulfillment of the auditability requirement" (Kaaniche and Laurent, 2017a; Laurent et al., 2018).
- approval of smart contracts creation— the service provider is requested to approve each smart contract creation by the data owner $\mathcal{O}$, by using its secret key $sk_{\mathcal{S}}$. More precisely, its secret key enables the service provider to decrypt the pairwise key $k_{\mathcal{C},\mathcal{S}}$ associated to the smart contract, and to prove its authenticity and its legitimacy to have a deciphered access to the shared data.

7.2.4. Availability

The blockchain relies on a highly decentralized infrastructure, thus providing our approach with availability assurance and liveness guarantees of data usage. We also point out the similarity between the well known double spending problem in bitcoin architectures (Karame et al., 2015) and the attack aiming at preventing a valid transaction to be registered in the blockchain. Indeed, both assume that an adversary has control over more than a half of the blockchain nodes, the achievement of which is assumed difficult (Karame et al., 2015).

8. CONCLUSION

Personal data are highly exposed to data leakage and misuse by third parties. As such, users have to own a complete control on their personal data usage without compromising their privacy or limiting service providers to propose personalized services and authorities' ability for auditing activities.

This paper introduces a blockchain-based solution for data usage auditing relying on both hierarchical ID-based encryption and signature mechanisms. Each data owner acts as a delegated PKG, and as such has the technical means to provide consent on his data usage and to control data collection and processing activities based on a per smart-contract approach, while enabling service providers to provide the evidence that any personal data processing was previously subjected to the consent by the data owner. Indeed, based on a consortium blockchain infrastructure, the proposed solution first enables the data owner to grant consent to service providers, specify their data access policy and track data usage flows in a trusted and privacy-preserving manner. Second, it provides a regulatory framework to properly enforce the legislation Regulation (EU) (2016). "For instance, in case of a non-compliance activity (i.e., unauthorized data access) reported by a data owner, authorized authorities may lead an investigation referring to as registered blockchain transactions with respect to concerned entities" (Laurent et al., 2018). Third, it helps in resolving availability concerns as blockchain transactions are replicated a large number of times on independent nodes.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation, to any qualified researcher.

AUTHOR CONTRIBUTIONS

NK did the main job. ML was closely supervising the research technical aspects. CL-B participated on the law aspects (GDPR).

FUNDING

This work was funded by the chair Policies and Values of Personal Information (cf. https://cvpip.wp.imt.fr/en/).

ACKNOWLEDGMENTS

This paper is an extended and revised version of our former conference work (Kaaniche and Laurent, 2017b) accepted in the 16th International Symposium on Network Computing and Applications (NCA), and is an extended discussion of Kaaniche and Laurent (2017c) with regard to smart contract design models and public vs. private blockchains.

The authors would like to thank the reviewer for their careful reading of the manuscript and their constructive remarks.

REFERENCES

Blazy, O., Kiltz, E., and Pan, J. (2014). "(Hierarchical) identity-based encryption from affine message authentication," in *International Cryptology Conference* (Berlin; Heidelberg: Springer), 408–425.

Chow, S. S., Hui, L. C., Yiu, S.-M., and Chow, K. (2004). "Secure hierarchical identity based signature and its application," in *ICICS, Vol. 4* (Malaga: Springer), 480–494.

Crosby, M., Pattanayak, P., Verma, S., and Kalyanaraman, V. (2016). Blockchain technology: beyond bitcoin. *Appl. Innov.* 2, 6–10.

Dierks, T., and Rescorla, E. (2008). *RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2*. Technical Report, IETF.

Dorri, A., Kanhere, S. S., Jurdak, R., and Gauravaram, P. (2017). LSB: A lightweight scalable blockchain for iot security and privacy. *arXiv[preprint]. arXiv:1712.02969*. doi: 10.1016/j.jpdc.2019.08.005

Fu, A., Yu, S., Zhang, Y., Wang, H., and Huang, C. (2017). NPP: A new privacy-aware public auditing sc chem for cloud data sharing with group users. *IEEE Trans. Big Data.* doi: 10.1109/TBDATA.2017.2701347

Gentry, C., and Halevi, S. (2009). "Hierarchical identity based encryption with polynomially many levels," in *Theory of Cryptography Conference* (San Francisco, CA: Springer), 437–456.

Gentry, C., and Silverberg, A. (2002). "Hierarchical ID-based cryptography," in *Advances in Cryptology–ASIACRYPT 2002*, 149–155.

Grumăzescu, C., Pura, M.-L., and Patriciu, V.-V. (2015). "Hybrid distributed-hierarchical identity based cryptographic scheme for wireless sensor networks," in *New Contributions in Information Systems and Technologies* (Azore: Springer), 949–958.

Horwitz, J., and Lynn, B. (2002). "Toward hierarchical identity-based encryption," in *Advances in Cryptology–EUROCRYPT 2002* (Amsterdam: Springer), 466–481.

Kaaniche, N., and Laurent, M. (2017a). "Attribute based encryption for multi-level access control policies," in *14th International Conference on Security and Cryptography, SECRYPT 2017* (Madrid).

Kaaniche, N., and Laurent, M. (2017b). "A blockchain-based data usage auditing architecture with enhanced privacy and availability," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)* (Cambridge, MA), 1–5.

Kaaniche, N., and Laurent, M. (2017c). *Blockchain Design Directions.* Internal Telecom SudParis Report.

Kaaniche, N., and Laurent, M. (2018). "BDUA: Blockchain-based data usage auditing," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)* (San Francisco, CA), 630–637.

Karame, G., Androulaki, E., Roeschlin, M., Gervais, A., and Capkun, S. (2015). Misbehavior in bitcoin: a study of double-spending and accountability. *ACM Trans. Inform. Syst. Secur.* 18:2. doi: 10.1145/2732196

Laurent, M. (2019). Authenticated and privacy-preserving consent management in the internet of things. *Proc. Comput. Sci.* 151, 256–263. doi: 10.1016/j.procs.2019.04.037

Laurent, M., Kaaniche, N., Le, C., and Vander Plaetse, M. (2018). "A blockchain-based access control scheme," in *15th International Conference on Security and Cryptography (SECRYPT)* (Porto), 168–176.

Lee, G. Y., Cha, K. J., and Kim, H. J. (2019). "Designing the gdpr compliant consent procedure for personal information collection in the iot environment," in *2019 IEEE International Congress on Internet of Things (ICIOT)* (Milan), 79–81.

Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., and Njilla, L. (2017). "Provchain: a blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (Madrid: IEEE Press), 468–477.

Linn, L., and Koo, M. (2016). *Blockchain for Health Data and Its Potential Use in Health IT and Health Care Related Research.* Available online at: https://www.healthit.gov/sites/default/files/11-74-ablockchainforhealthcare.pdf

Morel, V., Cunche, M., and Le Métayer, D. (2019). "A generic information and consent framework for the iot," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)* (Rotorua), 366–373.

Neisse, R., Steri, G., and Nai-Fovino, I. (2017). A blockchain-based approach for data accountability and provenance tracking. *arXiv[preprint]. arXiv:1706.04507*. doi: 10.1145/3098954.3098958

Ouaddah, A., Abou Elkalam, A., and Ait Ouahman, A. (2016). Fairaccess: a new blockchain-based access control framework for the internet of things. *Secur. Commun. Netw.* 9, 5943–5964.

Perng, G., Reiter, M. K., and Wang, C. (2005). "Censorship resistance revisited," in *Information Hiding* (Barcelona: Springer), 62–76.

Prabhakaran, M. M., and Sahai, A. (2013). *Secure Multi-Party Computation, Vol. 10*. IOS Press.

Ramachandran, A., and Kantarcioglu, M. (2018). "Smartprovenance: a distributed, blockchain based dataprovenance system," in *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy* (Tempe, AZ: ACM), 35–42.

Rantos, K., Drosatos, G., Kritsas, A., Ilioudis, C., Papanikolaou, A., and Filippidis, A. P. (2019). A blockchain-based platform for consent management of personal data processing in the IoT ecosystem. *Sec. Commn. Netw.* 2019:1431578. doi: 10.1155/2019/1431578

Regulation (EU) (2016). *2016/679 of the European Parliament and of the Council of 27 April 2016 on the Protection of Natural Persons With Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/ec (General Data Protection Regulation), ojeu l 119/1 of 4.05.2016*. Regulation (EU)

Seo, J. H., and Emura, K. (2015). "Revocable hierarchical identity-based encryption: history-free update, security against insiders, and short ciphertexts," in *Cryptographers Track at the RSA Conference* (San Francisco, CA: Springer), 106–123.

Shamir, A. (1985). "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in Cryptology* (New York, NY: Springer-Verlag New York, Inc.), 47–53.

Shetty, S., Red, V., Kamhoua, C., Kwiat, K., and Njilla, L. (2017). "Data provenance assurance in the cloud using blockchain," in *Disruptive Technologies in Sensors and Sensor Systems, Vol. 10206* (International Society for Optics and Photonics), 102060I.

Swan, M. (2015). *Blockchain: Blueprint for a New Economy*. O'Reilly Media, Inc, Sebastopol, CA.

Tian, M., and Huang, L. (2014). "Efficient identity-based signature from lattices," in *IFIP International Information Security Conference*, (Marrakesh: Springer), 321–329.

Tosh, D., Shetty, S., Foytik, P., Kamhoua, C., and Njilla, L. (2018). "Cloudpos: a proof-of-stake consensus design for blockchain integrated cloud," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, (San Francisco, CA).

Vukolic, M. (2017). "Rethinking permissioned blockchains," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts, BCC '17* (New York, NY: ACM), 3–7.

Wood, G. (2014). Ethereum: a secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 151. Available online at: https://ethereum.github.io/yellowpaper/paper.pdf

Yu, F. R., Tang, H., Mason, P. C., and Wang, F. (2010). A hierarchical identity based key management scheme in tactical mobile ad hoc networks. volume 7, pages 258–267. IEEE.

Zhang, Y., Lin, X., and Xu, C. (2018). "Blockchain-based secure data provenance for cloud storage," in *International Conference on Information and Communications Security* (Lille: Springer), 3–19.

Zyskind, G., Nathan, O., and Pentland, A. S. (2015). "Decentralizing privacy: using blockchain to protect personal data," in *Security and Privacy Workshops (SPW), 2015 IEEE* (San Jose, CA), 180–184.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.