# User-Generated Pseudonyms Through Merkle Trees

Georgios Kermezis[1], Konstantinos Limniotis[1,2](✉), and Nicholas Kolokotronis[3]

[1] School of Pure and Applied Sciences, Open University of Cyprus, 2220 Latsia, Cyprus
georgios.kermezis@st.ouc.ac.cy, konstantinos.limniotis@ouc.ac.cy
[2] Hellenic Data Protection Authority, Kifissias 1-3, 11523 Athens, Greece
klimniotis@dpa.gr
[3] Department of Informatics and Telecommunications, University of Peloponnese, Akadimaikou G.K. Vlachou Street, 22131 Tripolis, Greece
nkolok@uop.gr

**Abstract.** A pseudonymisation technique based on Merkle trees is described in this paper. More precisely, by exploiting inherent properties of the Merkle trees as cryptographic accumulators, we illustrate how user-generated pseudonyms can be constructed, without the need of a third party. Each such pseudonym, which depends on several user's identifiers, suffices to hide these original identifiers, whilst the unlinkability property between any two different pseudonyms for the same user is retained; at the same time, this pseudonymisation scheme allows the pseudonym owner to easily prove that she owns a pseudonym within a specific context, without revealing information on her original identifiers. Compared to other user-generated pseudonymisation techniques which utilize public key encryption algorithms, the new approach inherits the security properties of a Merkle tree, thus achieving post-quantum security.

**Keywords:** Data minimisation · General data protection regulation · Merkle trees · Personal data · Pseudonymisation

## 1 Introduction

Pseudonymisation of personal data constitutes an important privacy enhancing technique that, when appropriately implemented, suffices to provide specific data protection safeguards. More precisely, the data pseudonymisation may give rise to protecting (hiding) the real identities of the individuals (which is related, as a

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 786698. The work reflects only the authors' view and the Agency is not responsible for any use that may be made of the information it contains.

data protection goal, to data confidentiality), as well as to unlinkability of individuals across different application domains. Moreover, pseudonyms can also be used in some cases to ensure verification of the actual identity of the individual (which is related, as a data protection goal, to data integrity) [15,16]. Therefore, taking into account the six data protection goals as they have been presented in [19] for addressing the legal, technical, economic, and societal dimensions of privacy and data protection in complex IT systems - namely confidentiality, integrity, unlinkability, availability, intervenablity and transparency - the pseudonymisation may contribute in ensuring (at least) the three of them.

The aforementioned data protection goals of pseudonymisation are implied in the European General Data Protection Regulation (Regulation (EU) 2016/679 or GDPR). There are several references to pseudonymisation within the GDPR, mainly as the vehicle for providing appropriate data protection safeguards in several cases, such as towards achieving the so-called *data protection by design* principle. However, choosing a proper pseudonymisation technique is not always an easy task, since there are different parameters that need to be considered each time, taking into account the specific scenario that the pseudonymisation is to be used [15,16].

One quite challenging use case of pseudonymisation is the one that the user's pseudonym is being generated in the user's environment - i.e. user-generated pseudonyms. In such a scenario, neither the data controller (as is defined in the GPDR) nor any other (trusted or not) third party is actively employed in the process of deriving the pseudonyms; instead, the individuals by themselves, via a specific process in a decentralised approach, generate pseudonyms which in turn are being subsequently used by data controllers.

## 1.1 Related Work

Several pseudonymisation approaches focusing on deriving user - generated pseudonyms have been proposed in the literature (e.g. [24,34,36]), mainly based on public key cryptographic primitives. One of the most known scenarios of user-generated pseudonyms is the the case of several blockchain systems (such as the case of Bitcoin), in which the users are being identified by a meaningless identifier (i.e. the pseudonym, being called *address* in this case) which is uniquely associated with a relevant cryptographic key corresponding to its owner.

As stated in [24], when designing such a decentralised approach for pseudonym generation, we are mainly interested in fulfilling the following requirements: i) ease of use, ii) linking a pseudonym to its owning user should not be possible for any other than the user herself, unless it is explicitly permitted, iii) in cases that users may have multiple pseudonyms, it should not be possible to identify different pseudonyms as belonging to the same user, iv) injectivity, in terms that the pseudonym generation process should avoid duplicates, v) flexibility, i.e. it should be possible to add new pseudonyms to the user entities with minimal effort.

### 1.2 Contribution of This Work

In this paper, we explore the notion of the so-called cryptographic accumulators, in order to derive user-generated pseudonyms with some specific properties. Cryptographic accumulators are data structures based on cryptographic primitives to efficiently implement set membership operations [32]. They allow to accumulate a finite set of values $\{x_1, x_2, \ldots, x_n\}$ into a succinct value $X$. Therefore, they may constitute a convenient tool to derive pseudonyms (corresponding to the relevant succinct values) that are contingent on a set of initial identifiers (corresponding to the relevant set of values), so as to allow extracting the information whether a given identifier corresponds to a given pseudonym. To achieve this goal, we appropriately utilize the Merkle trees [28] - which is a case of a cryptographic accumulator - as the means to provide a new pseudonymisation technique. The generic idea of using Merkle trees for pseudonymisation purposes has been very recently discussed in [17]. In our approach, we propose a new pseudonymisation scheme such as, for a user $A$ with some domain-specific identifiers $\mathtt{id}_{A_0}, \ldots, \mathtt{id}_{A_{n-1}}$, a pseudonym $P_A$ of $A$ can be generated by the user $A$ herself, satisfying the following properties:

**P1.** The pseudonym $P_A$ depends on all $\mathtt{id}_{A_0}, \ldots, \mathtt{id}_{A_{n-1}}$.
**P2.** Knowledge of $P_A$ does not allow revealing any of the original identifiers $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, n-1$.
**P3.** The user $A$ can prove, whenever she wants, that any $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, n-1$, corresponds to $P_A$, without revealing any other information on the remaining identifiers $\mathtt{id}_{A_j}$, $j \in \{0, 1, \ldots, n-1\} \setminus \{i\}$.
**P4.** The user may generate several such pseudonyms $P_A^{(0)}, \ldots, P_A^{(s)}$, with the above properties, being pairwise unlinkable.
**P5.** Two different users $A$ and $B$ will always generate different pseudonyms $P_A$ and $P_B$, regardless the number, the types and the values of their original identifiers.

Therefore, the new pseudonymisation technique satisfies the properties described in [24] (implementation issues will be subsequently analysed), enriched with some additional properties that may be of high importance in specific scenarios, as discussed next. Actually, due to the property **P3**, the user $A$ may prove, if she wants, that two different pseudonyms $P_A^{(1)}$ and $P_A^{(2)}$, even if they have been given to two different organisations $\mathrm{Org}_1$ and $\mathrm{Org}_2$ respectively, correspond to her; such a proof of pseudonym's ownership though (i.e. proving to $\mathrm{Org}_1$ that she owns $P_A^{(2)}$ in $\mathrm{Org}_2$ and/or vice versa), does not reveal any additional information on the original identifiers of $A$ to either $\mathrm{Org}_1$ or $\mathrm{Org}_2$.

Moreover, it should be pointed out that the cryptographic strength of the above properties are strongly related to the cryptographic strength of the Merkle tree as an one-time signature scheme, which is known to be post-quantum secure under specific assumptions on the underlying hash function [9,10] (see Subsect. 3.3). This is an important property, taking also into account that other known techniques on deriving user-generated pseudonyms (such as the aforementioned techniques in [24,34,36])) rely on conventional public key cryptographic schemes which are not post-quantum secure.

The rest of the paper is organised as follows. First, the necessary background is given in Sect. 2, covering both the basic elements of the legal framework (Subsect. 2.1) and the typical Merkle trees (Subsect. 2.2). Next, the basic idea on the proposed pseudonymisation technique is presented in Sect. 3; this section also includes a discussion on the security properties and on implementation issues, as well as on possible application scenarios for this technique. Finally, concluding remarks are given in Sect. 4.

## 2 Preliminaries

### 2.1 Pseudonymisation and Data Protection: Legal Framework

The European Regulation (EU) 2016/679 (2016)—known as the *General Data Protection Regulation* or GDPR—constitutes the main legal instrument for personal data protection in Europe, which applies to all organizations that process personal data of individuals residing in the European Union, regardless of the organizations' location, which can be outside European Union.

The term *personal data* refers to any information relating to an identified or identifiable natural person, that is a person who can be identified (being called *data subject*). *Personal data processing* means any operation that is performed on personal data, including the collection, recording, structuring, storage, adaptation or alteration, retrieval, use, disclosure by transmission, dissemination, combination and erasure. The GDPR codifies the basic principles that need to be guaranteed when personal data are collected or further processed and sets specific obligations to the *data controllers* - i.e. the entities that, alone or jointly with others, determine the purposes and means of the processing of personal data. Amongst them, the so-called *data minimisation principle* refers to the necessity that the personal data shall be adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed (art. 5 of the GDPR).

The data minimisation, as a fundamental principle, spans the entire text of the GDPR: for example, is it explicitly mentioned in art. 25 towards ensuring the *data protection by design* principle, which in turn constitutes an important challenge involving various technological and organisational aspects [2]. Moreover, the art. 11 of the GDPR states that if the purposes for which the data controller processes personal data do not or do no longer require the identification of an individual, then the controller shall not be obliged to maintain, acquire or process additional information in order to identify the data subject.

In light of the above requirements, data pseudonymisation plays an important role in data protection. From an engineering perspective, a pseudonym is defined as an identifier of a subject, which is different from the subject's *real name* [1,33], whereas the types of pseudonyms may be distinguished by the context of use [33]. Typically, a pseudonym replaces a data subject's identifier, with the latter one being able to explicitly identify the data subject within a specific context;

for example, the original identifier can be a combination of first name and last name, an e-mail address, or even a device/network identifier (e.g. an IP address, a device ID etc.) which in turn constitute personal data when the device is associated with an individual (see also [12]).

The GDPR also defines pseudonymisation as *the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person*. As the GDPR explicitly states, pseudonymous data are personal and not anonymous data, despite the fact that there is often a confusion in characterizing pseudonymous data as anonymous (see, e.g., [12] for a discussion on this). However, with a properly implemented pseudonymisation scheme, pseudonymous data should not allow revealing the original identifier without some *additional information*; this piece of information can be, for example, a cryptographic key which is protected - and that's why a pseudonymisation technique often relies on utilizing a cryptographic function to identifiers or other identity-related information (see, e.g., [15,16] and the references therein).

## 2.2 Merkle Trees

A Merkle tree is a binary tree based on a cryptographic hash function $H$. Having as starting point $N$ values $y_0, \ldots, y_{N-1}$, where $N = 2^n$ for some integer $n$, the $i$-th leaf node is labeled with the corresponding hash value $H(y_i)$ of $y_i$, whereas every inner node is labeled with the hash value formed from the concatenation of its children's labels. The label of the root node is the accumulated value, which is clearly contingent on all $y_0, \ldots, y_{N-1}$. For example, in Fig. 1 which illustrates a Merkle tree of $2^3 = 8$ leaves (i.e. of height 3), it holds $a_{1,0} = H(a_{0,0} \parallel a_{0,1}) = H(H(y_0) \parallel H(y_1))$, $a_{2,0} = H(a_{1,0} \parallel a_{1,1})$ and $a_{3,0} = H(a_{2,0} \parallel a_{2,1})$.

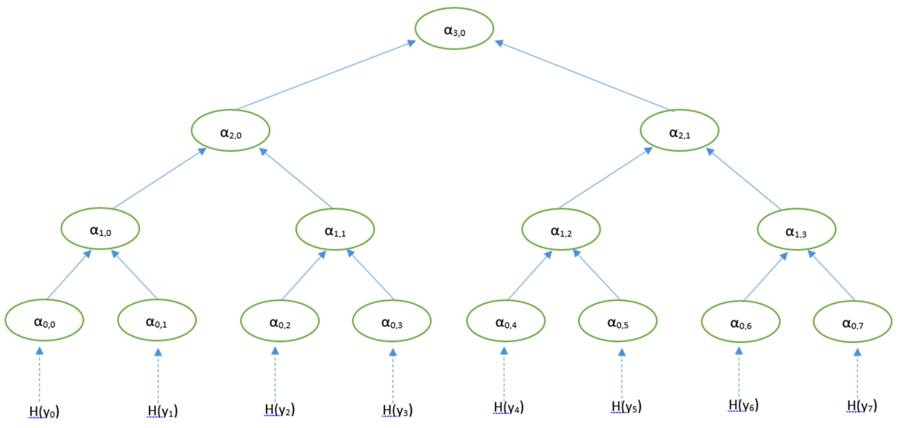

**Fig. 1.** A classical Merkle tree with $2^3$ leaves.

A Merkle tree may be the basis for deriving hash-based one-time signatures [28], generalizing the notion of Lamport's signatures [22]. Each $y_i$ corresponds to a private key that is being used to sign a message, whereas the root of the tree is the signer's public key. Let us assume, for simplicity, the simplest case in which each signature corresponds to a single-bit messages - i.e. either 0 or 1. When such a $y_i$ is to be used for signing one bit (either 0 or 1) of the message, the signer actually signs the message with the pair $(y_i, H(y_i))$ - i.e. the signer reveals the private value $y_i$ which allows for the computation of $H(y_i)$ which in turn is being used for the verification of the signature. More precisely, to allow signature verification, the signer should also reveal the *authentication path* of the tree - i.e. all the necessary labels of the intermediate nodes which are needed to verify the root of the tree, that is the signer's known public key. For example, for the case of the Merkle tree illustrated in Fig. 1, the verification path for the signature corresponding to $y_2$ consists of the labels $a_{0,3}$, $a_{1,0}$ and $a_{2,1}$ (see Fig. 2). Once a private key is being revealed it should not be used again for signing (and, thus, the number of possible messages that could be signed depends on the size of the tree). Actually, for any pair $y_i, y_{i+1}$, $i = 0, 2, 4, \ldots$, only one of these private values can be used for signing, depending on the message (i.e. $y_i$ is being used if the current bit of the message is 0 and $y_{i+1}$ if the current bit of the message is 1) and, after its usage, this pair is not being used for future signatures. The Merkle trees constitute the main building blocks for hash-based post-quantum secure signature schemes, such as LMS [26], XMSS [20], and SPHINCS+ [5].

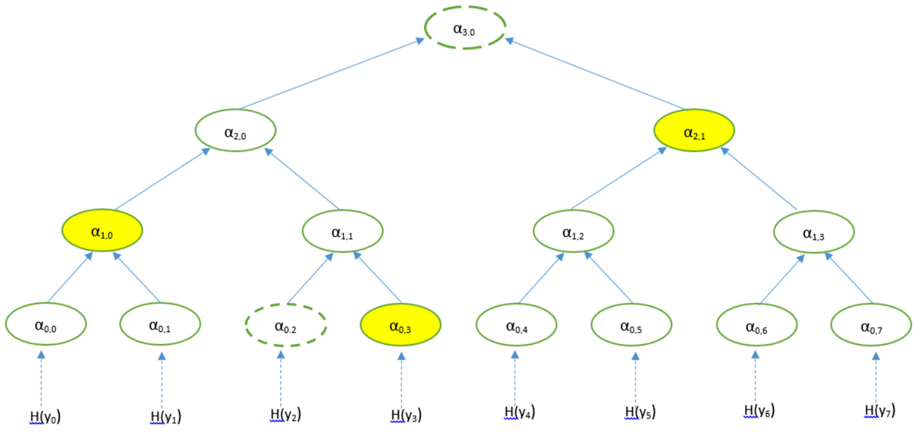

**Fig. 2.** An authentication path in a Merkle signature scheme.

## 3 The New Pseudonymisation Technique

In this section, we present how a Merkle tree can be used to generate pseudonyms with the properties **P1–P5** described in Sect. 1.

### 3.1 Initial Assumptions

Our main initial assumption is that an individual $A$ is being identified by different organisations $\text{Org}_0, \ldots, \text{Org}_{N-1}$ though pairwise distinct identifiers $\texttt{id}_{A_i}$, $i = 0, 1, \ldots, N-1$ respectively; for example, depending on the context, such identifiers could be. e.g., a Social Security Number, an Identity Card Number, a VAT Registration Number or even an e-mail address or a device identifier. We are not interested in how each $\text{Org}_i$, $i = 0, 1, \ldots, N-1$, obtains, at the first place, the identifier $\texttt{id}_{A_i}$; we simply assume that the validity of each such identifier, for the corresponding organisation, is ensured (i.e. through a secure registration procedure). This in general needs to be carefully considered - for example, it may need, depending on the context, a specific architectural structure based on recognised digital certificates; in any case, such an initial secure registration process is out of the scope of this paper.

Actually, the term *identifier* is being used hereinafter to describe any piece of information that allows distinguishing an individual from other individuals; the GDPR (Article 4) provides a non-exhaustive list of common identifiers (such as name, identification number, location data, online identifier). In our scenario, an identifier for each organisation could be of any form. However, our main assumption is that an organisation $\text{Org}_i$ should not, by default, be able to link personal information of $A$ with another organisation $\text{Org}_j$ (i.e. such a linking would vanish the data minimisation principle); this in turn means that, for any pair $i, j$, the organisation $\text{Org}_i$ should not know that $\texttt{id}_{A_j}$ is associated with the individual $A$. For example, a hospital may identify a patient though her Social Security Number; however, since there is no need for the hospital to get knowledge of the VAT Registration Number of the patient, this user's identifier should not become known to the hospital.

It should be pointed out that the notion of identifier is not necessarily restricted to a single value, as in the examples presented above; for example, one such identifier $\texttt{id}_{A_i}$ may consist, e.g., of a combination of individual's first name, last name and ID card number; although the ID card number by itself suffices to uniquely identify the individual in a specific context (i.e. it is a single identifier), we may allow incorporating many user's attributes into the definition of a single identifier. By these means, the number of possible single identifiers that may occur does not affect our analysis, since all of them may become part of a new single identifier incorporating all of them. Indeed, re-visiting the above example, the combination of first name and last name could be also an identifier (depending on the context), apart from the ID card number; however, we may define in this case that: $\texttt{id}_{A_1} = \text{first\_name} \parallel \text{last\_name} \parallel \text{ID\_card\_number}$.

Moreover, in the subsequent analysis, we assume that the organisations $\text{Org}_i$, $i = 0, 1, \ldots, N-1$ do not secretly exchange information towards getting, for a user $A$, much more personal information of $A$ than they need for their data processing purposes (i.e. they do not collaborate maliciously in terms of personal data protection requirements). In other words, for any $i \neq j$, $\text{Org}_i$ and $\text{Org}_j$ do not try to link information relating to $\texttt{id}_{A_i}$ and $\texttt{id}_{A_j}$ - which would also lead in revealing $\texttt{id}_{A_i}$ (resp. $\texttt{id}_{A_j}$) to $\text{Org}_j$ (resp. $\text{Org}_i$). It should be noted that

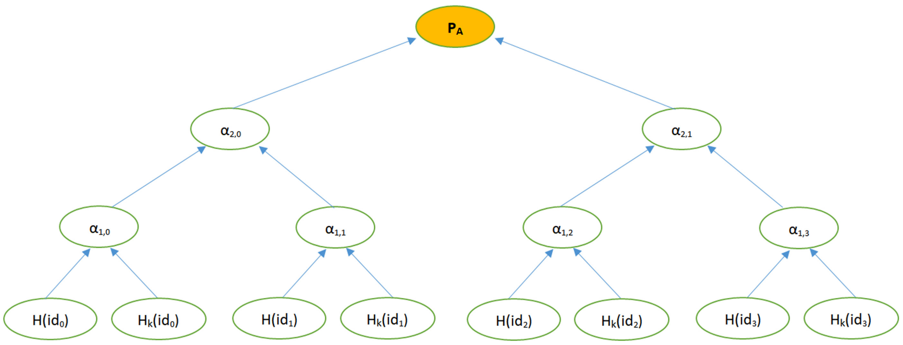

**Fig. 3.** Generating a pseudomyn depending on $\mathtt{id}_{A_0}, \mathtt{id}_{A_1}, \mathtt{id}_{A_2}, \mathtt{id}_{A_3}$

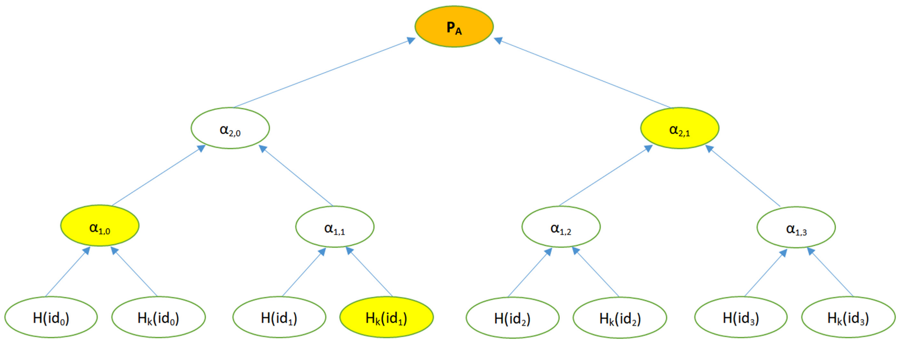

**Fig. 4.** Proving that the pseudonym $P_A$ corresponds to the known identifier $\mathtt{id}_{A_1}$

such linking attacks, generally, may be possible between different databases, even if the relevant users identifiers are different: this may occur, for example, by comparing the so-called quasi-identifiers of individuals (see, e.g., [18]). In our paper, we do not consider such threats; we simply focus on deriving user-generated different pseudonyms $P_A^{(i)}$ and $P_A^{(j)}$ being pairwise unlinkable - i.e., for any $i \neq j$, $P_A^{(i)} \neq P_A^{(j)}$, whilst knowledge of $P_A^{(i)}$, $i = 0, 1, \ldots, N-1$, does not allow computation of $\mathtt{id}_{A_i}$, as well as computation of any other identifier $\mathtt{id}_{A_j}$ of $A$. Moreover, under the aforementioned assumption of the honest model with respect to the behavior of the parties (organisations), there is no way to establish that $P_A^{(i)}$ and $P_A^{(j)}$ correspond to the same individual - unless the individual $A$ by herself wants to prove it.

## 3.2 How to Derive a Pseudonym

Let us assume that the user $A$ wants to generate $N$ pseudonyms $P_A^{(i)}$, $i = 0, 1, \ldots, N-1$, each for one different organisation $\mathrm{Org}_i$, where $\mathrm{Org}_i$ already knows the identifier $\mathtt{id}_{A_i}$ of $A$. First, for the sake of simplicity, we assume that

$N = 2^n$ for some integer $n$ (and the more general case will be subsequently discussed). For these parameters, each pseudonym is associated with a Merkle tree of height $n+1$ - i.e. with $2N$ leaves. For the $i$-th such Merkle tree, the labels $a_{0,0}^{(i)}, a_{0,1}^{(i)}, \ldots, a_{0,2N-1}^{(i)}$ of its leaves are constructed via a secret key $k^{(i)}$, which is known only to $A$, as follows:

$$a_{0,\ell}^{(i)} = \begin{cases} H(\mathtt{id}_{A_{\ell/2}}), & \text{if } \ell = 0, 2, 4, \ldots, 2N-2, \\ H_{k^{(i)}}(\mathtt{id}_{A_{(\ell-1)/2}}), & \text{if } \ell = 1, 3, 5, \ldots, 2N-1 \end{cases}$$

where $H$ is a cryptographic hash function and $H_{k^{(i)}}$ is a keyed hash function with a key $k^{(i)}$ - i.e. a Message Authentication Code (MAC). Actually, in simple words, each pair of leaves of the Merkle tree corresponds to one users's identifier; the first leaf of the pair is labelled by the hashed value of the identifier, whilst the second leaf is labelled by the keyed hashed value of the identifier (i.e. its MAC). Such a Merkle tree is shown in the Fig. 3, with 8 leaves (i.e. $N = 4$); the label $P_A$ of the root corresponds to the pseudonym of the user (for simplicity, we write in this Figure $P_A$ and $k$ instead of $P_A^{(i)}$ and $k^{(i)}$ respectively). Apparently, the root of the tree depends on all its leaves - namely, on all identifiers $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, N-1$. Moreover, by using different key $k^{(i)}$ for each Merkle tree, a different pseudonym $P_A^{(i)}$ is being generated for each Org$_i$; all of these pseudonyms $P_A^{(i)}$ though, $i = 0, 1, \ldots, N-1$, share the same properties (i.e. each of them depends on all $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, N-1$,) and are pairwise unlinkable.

Since we assume that these pseudonyms are being generated in the user's environment, which in turn means that each secret key $k^{(i)}$ is known only to the pseudonym's owner $A$, it is essential to establish a procedure that the user $A$ proves to an organisation Org$_j$ that she owns the pseudonym $P_A^{(i)}$. Recalling our assumption that Org$_j$ already knows $\mathtt{id}_{A_j}$ as an identifier of $A$, the properties of a Merkle tree as a cryptographic primitive for a digital signature (see Fig. 2) suffice to allow such a proof of ownership. Indeed, let us assume that, for the pseudonym derived in the Fig. 3, that the user $A$ wants to prove to Org$_1$ that she owns this pseudonym. Then $A$ simply reveals the labels of the authentication path of the tree that allows Org$_1$ verify that the pseudonym $P_A^{(i)}$ corresponds to $\mathtt{id}_{A_1}$ - namely, $A$ reveals the values $H_{k^{(i)}}(\mathtt{id}_{A_1})$, $a_{1,0}$ and $a_{2,1}$ (recall that Org$_1$ can compute $H(\mathtt{id}_{A_1})$). This is illustrated in Fig. 4. Clearly, a similar procedure may be followed for any identifier $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, N-1$ or, equivalently, to any organisation Org$_i$, $i = 0, 1, \ldots, N-1$. Due to the properties of the Merkle tree, no information on other identifiers of $A$ is revealed in this verification procedure. Moreover, for a secure keyed hash function, knowledge of the pair $(\mathtt{id}_{A_i}, H_{k^{(i)}}(\mathtt{id}_{A_i}))$ does not allow computing $k^{(i)}$.

An interesting remark is that the above verification procedure holds for any pair $i, j \in \{0, 1, \ldots, N-1\}$ - i.e. $A$ can always prove to Org$_j$ that $P_A^{(i)}$ corresponds to $\mathtt{id}_{A_j}$, without revealing to Org$_j$ any other information on the remaining identifiers $\mathtt{id}_{A_\ell}$, $\ell \in \{0, 1, \ldots, N-1\} \setminus \{j\}$. If $i = j$, such a verification procedure is essential in establishing that the pseudonym of $A$ within Org$_j$ will be $P_A^{(j)}$; this is a necessary first step (i.e. a registration process), since a pseudonym $P_A^{(j)}$ may be considered as valid in a specific context only if the relevant organisation $\mathrm{Org}_j$ that will use $P_A^{(j)}$ is ensured for its validity. However, if $i \neq j$, then we are in the case that $A$ proves to $\mathrm{Org}_j$ that she owns another pseudonym $P_A^{(i)}$ in another organisation $\mathrm{Org}_i$ - i.e. the user $A$ allows for linking personal information of herself. Note that, without this intervention (i.e. proof of ownership) of $A$, such a linking between $P_A^{(i)}$ and $P_A^{(j)}$ is not possible.

Due to the nature of the Merkle tree, which is typically considered as a full balanced binary tree, the number of its leaves is always a power of 2; that's why we first assumed that $N = 2^n$ for some $n$. However, this may not be always the case. To alleviate this issue, we choose the minimum possible integer $m$ such that $N < 2^m$ and we proceed accordingly by constructing a Merkle tree with $2^{m+1}$ leaves; the first $2N$ of them correspond to the $N$ user's identifiers as described previously, whereas the remaining $2^{m+1} - 2N$ leaves can be chosen arbitrarily.

## 3.3 Security Requirements

As in any hash-based post-quantum signature scheme, the security of the described pseudonymisation scheme rests with the cryptographic properties of the underlying hash function $H$ in the Merkle tree [10,11]. In our case, the necessary security requirement for $H$ is collision resistance, which means that finding two inputs with the same hash value is computationally infeasible. Collision resistance implies other weaker security requirements such as one-wayness (i.e. for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output) and second-preimage resistance (i.e. it is computationally infeasible to find a second input which has the same hash value as that of a specified input). Therefore, if collision resistance is present, an adversary cannot find, for any given pseudonym $P_A^{(i)}$, as well as for any given label of an intermediate node of the tree, any relevant $\mathtt{id}_{A_\ell}$, $\ell = 0, 1, \ldots, N-1$. Moreover, an adversary cannot find/compute inputs which give rise to the same outputs as the original $\mathtt{id}_{A_\ell}$, $\ell = 0, 1, \ldots, N-1$.

An important parameter of hash functions towards evaluating the aforementioned collision resistance is the length $n$ of the hash value. As it is stated in [10], the so-called birthday attack, which works for any hash function, finds a collision in time approximately $2^{n/2}$; moreover, in a quantum world, there exists an attack finding a collision in time approximately $2^{n/3}$ [4]. Therefore, today we need $n \geq 256$, whilst for post-quantum security we need $n \geq 384$ [10]. The cryptographic NIST standards SHA-2 [30] and SHA-3 [31] are known to be collision-resistant, also supporting such lengths for their output.

Our pseudonymisation scheme also utilises a keyed hash function. Apparently, the requirement of collision resistance is also necessary for this function. To this end, the HMAC [29] or GMAC [13] cryptographic standards can be, e.g., used. For the case of HMAC, it is well-known [6] that its cryptographic strength depends on the properties of the underlying hash function - whereas the security of HMAC is formally studied in [7]. Therefore, again an appropriate collision resistant hash function, such as SHA-2 or SHA-3, suffices to develop a secure keyed hash function (see also ENISA's report on cryptographic parameters [14]). The security properties of the GMAC, being an information-theoretic authentication code, already assumes an attacker with unlimited computing power and, thus, it provides protection in the post-quantum era (see, e.g., [3]). Regarding the key size, GMAC provides post-quantum security with 128 bits key size [3]. The same key size is also considered as adequate today for HMAC; however, although there is a belief that such a key size will also be adequate for HMAC in the post-quantum era, several post-quantum implementations of HMAC utilize larger key sizes (see, e.g., [23]).

Apart from the properties of the cryptographic primitives that are being used, the following security requirement should also necessarily be in place: When the user $A$ registers her pseudonym $P_A^{(i)}$ to the organisation $\mathrm{Org}_i$, this registration should be authenticated (i.e. the identity of $A$ providing $\mathtt{id}_{A_i}$ shall be ensured). Otherwise, an adversary having knowledge of the original identifier $\mathtt{id}_{A_i}$ of $A$ could clearly create a fake pseudonym $P_A^{(i')}$ for which he could prove that it stems from $\mathtt{id}_{A_i}$.

In a similar manner, the authentication path needs also to be fully protected; indeed, if an authentication path for a specific identifier is leaked, then this can be subsequently used by an adversary in a type of replay-attack (similarly to the case of post-quantum digital signatures based on Merkle trees, which do not allow the usage of the same authentication path twice for signing). In the same direction, it should be pointed out that the proposed scheme does not alleviate the potential threat of users acting maliciously, in terms of sharing their identifiers and authentication paths with other users (this is an issue that is being discussed in [25] as an important concern for some applications). In any case though, it is logical to assume that, in specific applications, the users do not want to share such information because they do need to protect their personal data.

### 3.4 Possible Applications

Based on the above properties of the pseudonymisation scheme, several possible application scenarios can be considered, in which the proposed pseudonymisation scheme provides the means to ensure data minimisation. Some indicative applications are given below, actually based on the properties **P1**–**P5** of the scheme.

**Minimizing Information Requested from Another Entity/Data Controller.** Let us assume that the individual $A$ is identified by $\mathtt{id}_{A_1}$ in $\mathrm{Org}_1$ and by $\mathtt{id}_{A_2}$ in $\mathrm{Org}_2$, whereas the organisation $\mathrm{Org}_1$ needs to get some information about $A$ from $\mathrm{Org}_2$. However, $\mathrm{Org}_1$ (resp. $\mathrm{Org}_2$) should not get knowledge of $\mathtt{id}_{A_2}$ (resp. $\mathrm{Org}_1$).

As a specific indicative example, $\mathrm{Org}_1$ could be, e.g., a University, using the student number of $A$ as $\mathtt{id}_{A_1}$, whereas $\mathrm{Org}_2$ could be the Finance Ministry, with the Department/Service that is responsible for citizens taxes, using the VAT number as $\mathtt{id}_{A_2}$. Let us assume that the University needs to get knowledge on the annual income of the student $A$, in order to decide whether the student $A$ deserves some benefits (e.g. housing alliance). Actually, due to the data minimisation principle, if the decision on students benefits is based on a threshold of their annual income, only knowledge on whether this income is higher or lower than the threshold is needed (and not the exact value of the income).

With our pseudonymisation scheme, the user $A$ has already created the pseudonyms $P_A^{(1)}$ and $P_A^{(2)}$ for the University and Finance Ministry respectively, having proved to each of them their ownership. Each of these pseudonyms is the root of a Merkle tree with 4 leaves; the leaves, in both trees, corresponds to the student number and the VAT number. To allow the University obtain the necessary information, the student $A$ may provide to her University her pseudonym $P_A^{(2)}$ that uses for the service of the Finance Ministry, proving also that this pseudonym indeed corresponds to her; note that, for such a proof of ownership, $A$ will provide to the University the authentication path of the Merkle tree of $P_A^{(2)}$ corresponding to her student number and not to the VAT Number (i.e. a different authentication path from the one that $A$ used to prove the ownership of $P_A^{(2)}$ to the Finance Ministry). In other words, the University is able to verify the ownership of $P_A^{(2)}$, without getting any information on the VAT number of $A$. Then, the University simply asks the Finance Ministry to response whether the annual income of the individual with the pseudonym $P_A^{(2)}$ if higher or lower than the prescribed threshold. By these means, the University receives the minimum possible information that is necessary to perform its tasks.

Clearly, other similar examples as the above may be considered.

**Minimizing Exchange of Information Between Joint Data Controllers or Between a Data Controller and a Data Processor.** There are also cases in which a specific personal data processing is being somehow shared between different entities - each of them having a specific role in the whole process. Depending on the role, this could be a case of joint controllership (i.e. two or more entities are joint data controllers, that is they jointly determine the purposes and means of processing) or the case that some entities are data processors (i.e. entities which process personal data on behalf of the controller(s)). In any case, there may be necessary that such a single entity should have restricted access to personal data, in the framework of the data minimisation principle. For example, let us assume that one entity (let's say $\mathrm{Org}_2$) may perform analysis on raw data, so as to derive an output for each individual which in turn will be feed to another entity (let's say $\mathrm{Org}_1$). It is probable that, due to the data minimisation principle, $\mathrm{Org}_2$ should not be able to re-identify the individuals, whilst $\mathrm{Org}_1$, being able to re-identify them, should get access only to the outcome of the processing of $\mathrm{Org}_2$ and not to the whole volume of initial raw data.

As a possible application scenario lying in this case, we may refer to data processing based on the Pay-How-You-Drive model. The main idea of this insurance model is that drivers have to pay a premium based on their driving behaviour and degree of exposure, instead of paying a predetermined fixed price. Such a model poses several privacy risks, due to the fact that the evaluation of the driver's behavior typically necessitates tracking of the driver's routes, collecting and/or extracting detailed personal information (speed, harsh braking/acceleration, visited places, trips frequencies and time schedules, number and duration of possible stops etc.). Hence, the proposed pseudonymisation scheme could possibly alleviate such privacy threats as follows:

- The collection of raw data, based on the driver's driving information, is performed by $\text{Org}_2$, which in turn performs the whole analysis in order to derive a scoring for the driver (according to the model). The scoring by itself does not allow going backwards to the detailed personal information.
- $\text{Org}_2$ works on pseudonymised information. For an individual (driver) $A$, $\text{Org}_2$ uses a user-generated pseudonym $P_A^{(2)}$ based on an identifier $\texttt{id}_{A_2}$ of $A$ which is known only to $\text{Org}_2$. Although it is important that $\texttt{id}_{A_2}$ is unique for $A$ and suffices to discriminate her from any other user, the value $\texttt{id}_{A_2}$ by itself should not be able to allow finding the identity of $A$. For example, $\texttt{id}_{A_2}$ could be a unique identifier generated by the relevant smart application.
- $\text{Org}_2$ submits the output of its analysis (i.e. scoring of the driver) to the insurance company $\text{Org}_1$, in a pseudonymised form, based on the pseudonym $P_A^{(2)}$. Note that, at this moment, $\text{Org}_1$ is not able to link $P_A^{(2)}$ to any of its insured customers (whereas even $\texttt{id}_{A_2}$ is not known to $\text{Org}_1$).
- $\text{Org}_2$ deletes the raw data.
- The user $A$ proves to the $\text{Org}_1$ that she owns the pseudonym $P_A^{(2)}$. Such a proof of ownership is based on an identifier $\texttt{id}_{A_1}$, which is known only to $\text{Org}_1$ for identifying its customers, which had been also used, in conjunction with $\texttt{id}_{A_2}$, for constructing $P_A^{(2)}$. In other words, $P_A^{(2)}$ is the root of a Merkle tree, whose four leaves are based on two identifiers $\texttt{id}_{A_1}$ and $\texttt{id}_{A_2}$ - namely, they are being labelled by $\texttt{id}_{A_1}$, $H_k(\texttt{id}_{A_1})$,$\texttt{id}_{A_2}$, $H_k(\texttt{id}_{A_2})$ respectively.
- The user (driver) may create new pseudonyms (i.e. by changing her secret keys) for a new relevant processing by $\text{Org}_1$ and $\text{Org}_2$.

By the above procedure, the insurance company gets the desired information on the evaluation of the driving behavior of its customer $A$, without obtaining any detailed personal information of her driving habits (trips, speed etc.). Moreover, $\text{Org}_2$ which get (and processes) such detailed personal information, is not able to link this information to an identified individual, unless of course the user $A$ wants to prove her exact identity to $\text{Org}_2$ - e.g. in case of disputing the accuracy of personal data processed (recall also our assumption with respect to the honest model, in the beginning of Sect. 3).

Although the above clearly does not constitute a full solution to the privacy challenges of a Pay-How-You-Drive model, it becomes clear that by utilizing advanced pseudonymisation techniques such the one presented here, more options for alleviating data protection issues in several application scenarios are present.

## 3.5 Implementation Analysis

The performance of Merkle trees in terms of one-time signature schemes, taking into account both speed and storage, has been widely discussed by several researchers, whereas several improvements deviating from the naive implementation have been also proposed. Namely, there are techniques aiming to generate the signature without saving too many nodes, at a still efficient time - i.e. dealing with the so-called Merkle tree traversal problem (see, e.g., [8,27,35]).

Here, we do not focus on finding the best possible implementation, but we rather discuss several aspects that need to be considered. First, the height of the tree depends fully on the number of leaves - which, in our case, is fully determined by the number $N$ of original user's identifiers $\mathtt{id}_{A_i}$, $i = 0, 1, \ldots, N-1$. The height in turn determines the total number of the tree's nodes, as well as the length of the verification path (which in general affects network performance). More precisely, in a Merkle tree of height $h$, the verification path also includes $h$ nodes. For example, if $h = 4$, which means that the number of user's domain-specific identifiers is $N = 8$, and the size of the hash value is 384 bits (i.e. to be consistent with the requirements for post-quantum security), then the size of the verification path, to verify the pseudonym of length 384 bits, is 192 Bytes.

Moreover, as stated above, since the typical form of a Merkle tree implies a balanced tree (i.e. all leaves reside at the bottom level $h$, where $h$ is the height of the tree), the number of the leaves should be a power of 2, and this forms a restriction of the whole process (it is already discussed that utilizing dummy values could be a way to deal with this issue). Alternatively, this could be alleviated by extending the notion of the Merkle tree so as to be imbalanced (see, e.g., [21]); in such a scenario, no all verification paths will have the same length. In any case though, adding a new leaf (i.e. adding a new identifier) yields a new root of the Merkle tree - i.e. a new pseudonym - and, therefore, an extended verification path. We state the above as possible approaches that could be considered as future research steps.

In any case, it is expected that the utilization of a Merkle tree as a vehicle for pseudonymisation through the proposed approach will not lead to a large tree, since typically the number of individual's identifiers will not be too large (see, e.g., the discussion of possible application areas in Subsect. 3.4). To verify the effectiveness of the approach for small sizes of Merkle trees for $N$ identifiers or, equivalently, $2N$ leaves, $2 \leq N \leq 128$, we executed experiments in a typical Windows 10 home computer (AMD Ryzen 3 2200U with Radeon Vega Mobile Gfx 2.5 GHz, with 8 Gb RAM). We utilized Python v.3.8.2 for our implementation of the pseudonymisation scheme, whilst we measured both the time for creating the pseudonym (i.e. to create the Merkle tree) as well as the time for the pseudonym verification (i.e. to verify that the given verification path does yield the pseudonym, for the given identifier). We utilized the SHA-2 hash function, with hash length 256 bits, and the HMAC, via the relevant Python libraries. The results illustrate that, for $N \leq 128$, the time for creating a Merkle tree (without implementing the most effective approach) is less than 1 s (starting from about

15 ms for small values of $N$), whereas the time for pseudonym verification is less than 40 ms.

## 4 Conclusion and Future Work

In this paper, we illustrated how a Merkle tree (as a characteristic example of cryptographic accumulator) can be used to provide a pseudonymisation mechanism with specific nice properties in terms of fulfilling the data minimisation principle, which are not easily attained by other known pseudonymisation techniques. Moreover, since the Merkle tree is known to provide post-quantum security, the proposed pseudonymisation scheme suffices to provide long-term security, whereas other known cryptography-based pseudonymisation techniques with similar properties do not provide such post-quantum resistance. The main outcome of the above analysis is that advanced cryptography suffices to provide solutions to personal data protection matters - not only from a security perspective (which is the obvious one) but also from a data minimisation perspective.

The ideas presented in this paper opens several directions for further research. First, for any specific case study utilizing this pseudonymisation scheme, a formal security analysis of the whole procedure is essential to be performed. Furthermore, other extensions of the Merkle trees (as in the cases of post-quantum one time signatures like XMSS or SPHINCS) could be possibly studied in the framework of deriving pseudonymisation schemes. Another interesting direction is to examine how the pseudonym's owner will be able to explicitly define which types of personal data will be exchanged between the two entities (organisations), so as to eliminate the risk that the two organisations exchange more personal data than it is necessary (i.e. to force transparency on the data exchange to the user).

In any case, a general conclusion is that the properties achieved by the proposed technique should be further elaborated, in terms of identifying other application areas in which this technique could be beneficial.

**Acknowledgment.** The authors would like to thank the anonymous reviewers for their very useful comments which helped to improve the paper.

## References

1. Akil, M., Islami, L., Fischer-Hübner, S., Martucci, L.A., Zuccato, A.: Privacy-preserving identifiers for IoT: a systematic literature review. IEEE Access **8**, 168470–168485 (2020). https://doi.org/10.1109/ACCESS.2020.3023659
2. Alshammari, M., Simpson, A.: Towards a principled approach for engineering privacy by design. In: Schweighofer, E., Leitold, H., Mitrakas, A., Rannenberg, K. (eds.) APF 2017. LNCS, vol. 10518, pp. 161–177. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67280-9_9
3. Bernstein, D.J., Lange, T.: Post-quantum cryptography. Nature **549**, 188–194 (2017). https://doi.org/10.1038/nature23461
4. Brassard, G., Høyer, P., Tapp, A.: Quantum algorithm for the collision problem. In: Kao, M.Y. (ed.) Encyclopedia of Algorithms. Springer, New York (2016). https://doi.org/10.1007/978-1-4939-2864-4_304

5. Aumasson, J.-P., et al.: SPHINCS+ - submission to the 2nd round of the NIST post-quantum project. Specificatin document (2019). https://sphincs.org/data/sphincs+-round2-specification.pdf
6. Bellare, M., Canetti, R., Krawczyk, H.: Keying hash functions for message authentication. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 1–15. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_1
7. Bellare, M.: New proofs for NMAC and HMAC: security without collision-resistance. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 602–619. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_36
8. Berman, P., Karpinski, M., Nekrich, Y.: Optimal trade-off for Merkle tree traversal. Theor. Comput. Sci. **372**(1), 22–36 (2007). https://doi.org/10.1016/j.tcs.2006.11.029
9. Buchmann, J., Dahmen, E., Hülsing, A.: XMSS - a practical forward secure signature scheme based on minimal security assumptions. PQCrypto 2011: Post-Quantum Cryptography, pp. 117–129 (2011)
10. Buchmann, J.A., Butin, D., Göpfert, F., Petzoldt, A.: Post-quantum cryptography: state of the art. In: Ryan, P.Y.A., Naccache, D., Quisquater, J.-J. (eds.) The New Codebreakers. LNCS, vol. 9100, pp. 88–108. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49301-4_6
11. Buchmann, J., García, L.C.C., Dahmen, E., Döring, M., Klintsevich, E.: CMSS – an improved Merkle signature scheme. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 349–363. Springer, Heidelberg (2006). https://doi.org/10.1007/11941378_25
12. Chatzistefanou, V., Limniotis, K.: On the (non-)anonymity of anonymous social networks. In: Katsikas, S.K., Zorkadis, V. (eds.) e-Democracy 2017. CCIS, vol. 792, pp. 153–168. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71117-1_11
13. Dworkin, M.: Recommendation for block cipher modes of operation: galois/counter mode (GCM) and GMAC. NIST Special Publication 800–38D (2007)
14. European Union Agency for Cybersecurity: Algorithms, key sizeand parameters report (2014). https://doi.org/10.2824/36822
15. European Union Agency for Cybersecurity: Recommendations on shaping technology according to GDPR provisions - an overview on data pseudonymisation (2018). https://doi.org/10.2824/74954
16. European Union Agency for Cybersecurity: Pseudonymisation Techniques and Best Practices (2019). https://doi.org/10.2824/247711
17. European Union Agency for Cybersecurity: Data Pseudonymisation: Advanced Techniques and use cases (2021). https://doi.org/10.2824/860099
18. Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: a survey of recent developments. ACM Comput. Surv. **42**, Article 14 (2010). https://doi.org/10.1145/1749603.1749605
19. Hansen, M., Jensen, M., Rost, M.: Protection goals for privacy engineering. In Proceedings of the 2015 IEEE Security and Privacy Workshops (SPW 2015), pp. 159–166. IEEE (2015). https://doi.org/10.1109/SPW.2015.13
20. Huelsing, A., Butin, D. Gazdag, S.-L., Rijneveld, J., Mohaisen, A.: XMSS: eXtended Merkle Signature Scheme. RFC 8391 (2018). https://rfc-editor.org/rfc/rfc8391.txt

21. Kandappu, T., Sivaraman, V., Boreli, R.: A novel unbalanced tree structure for low-cost authentication of streaming content on mobile and sensor devices. In: 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), Seoul, pp. 488–496 (2012). https://doi.org/10.1109/SECON.2012.6275816
22. Lamport, L.: Constructing digital signatures from a one way function. Technical report SRI-CSL-98, SRI International Computer Science Laboratory (1979)
23. Latif, M.K., Jacinto, H.S., Daoud, L., Rafla, N.: Optimization of a quantum-secure sponge-based hash message authentication protocol. In: 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS), Windsor, Canada, pp. 984–987 (2018). https://doi.org/10.1109/MWSCAS.2018.8623880
24. Lehnhardt, J., Spalka, A.: Decentralized generation of multiple, uncorrelatable pseudonyms without trusted third parties. In: Furnell, S., Lambrinoudakis, C., Pernul, G. (eds.) TrustBus 2011. LNCS, vol. 6863, pp. 113–124. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22890-2_10
25. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: Heys, H., Adams, C. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-46513-8_14
26. McGrew, D., Curcio, M., Fluhrer, S.: Leighton-Micali hash-based signatures. RFC 8554 (2019). https://rfc-editor.org/rfc/rfc8554.txt
27. Merkle, R.C.: A digital signature based on a conventional encryption function. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 369–378. Springer, Heidelberg (1988). https://doi.org/10.1007/3-540-48184-2_32
28. Merkle, R.C.: A certified digital signature. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 218–238. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_21
29. National Institute of Standards and Technology: The Keyed-Hash Message Authentication Code (HMAC). FIPS PUB 198–1 (2008). https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.198-1.pdf
30. National Institute of Standards and Technology: Secure Hash Standard (SHS). FIPS PUB 80-4 (2015). https://doi.org/10.6028/NIST.FIPS.180-4
31. National Institute of Standards and Technology: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. FIPS PUB 202 (2015). https://doi.org/10.6028/NIST.FIPS.202
32. Ozcelik, I., Medury, S., Broaddus, J., Skjellum, A.: An overview of cryptographic accumulators. In: 7th International Conference on Information Systems Security and Privacy (ICISSP 2021), pp. 661–669 (2021)
33. Pfitzmann, A., Hansen, M.: A terminology for talking about privacy by data minimization: anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. TU Dresden, Dresden Germany, Technical report V0.34 (2010)
34. Schartner, P., Schaffer, M.: Unique user-generated digital pseudonyms. In: Gorodetsky, V., Kotenko, I., Skormin, V. (eds.) MMM-ACNS 2005. LNCS, vol. 3685, pp. 194–205. Springer, Heidelberg (2005). https://doi.org/10.1007/11560326_15
35. Szydlo, M.: Merkle tree traversal in log space and time. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 541–554. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_32
36. Tunaru, I., Denis, B, Uguen, B.: Location-based pseudonyms for identity reinforcement in wireless ad hoc networks. In: Proceedings of IEEE 81st Vehicular Technology Conference (VTC Spring), pp. 1–5 (2015). https://doi.org/10.1109/VTCSpring.2015.7145918