IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

*Digital Object Identifier 10.1109/ACCESS.2017.DOI*

# Burnable Pseudo-Identity: A Non-Binding Anonymous Identity Method for Ethereum

**IVÁN GUTIÉRREZ-AGÜERO[1], XABIER LARRUCEA[1], (SENIOR MEMBER, IEEE), SERGIO ANGUITA.[1], AITOR GOMEZ-GOIRI.[1], BORJA URQUIZU.[1],**
[1]Tecnalia, Basque Research and Technology Alliance (BRTA), 48170 Derio, Spain

Corresponding author: Xabier Larrucea (e-mail: Xabier.larrucea@ tecnalia.com).

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 786713 (POSEIDONproject).

**ABSTRACT** The concept of identity has become one common research topic in security and privacy where the real identity of users must be preserved, usually covered by pseudonym identifiers. With the rise of Blockchain-based systems, identities are becoming even more critical than before, mainly due to the immutability property. In fact, many publicly accessible Blockchain networks like Ethereum rely on pseudonymization as a method for identifying subject actions. Pseudonyms are often employed to maintain anonymity, but true anonymity requires unlinkability. Without this property, any attacker can examine the messages sent by a specific pseudonym and learn new information about the holder of this pseudonym. This use of Blockchain collides with regulations because of the right to be forgotten, and Blockchain-based solutions are ensuring that every data stored within the chain will not be modified. In this paper we define a method and a tool for dealing with digital identities within Blockchain environments that are compliant with regulations. The proposed method provides a way to grant digital pseudo identities unlinked to the real identity. This new method uses the benefits of key derivation systems to ensure a non-binding interaction between users and the information model associated with their identity. The proposed method is demonstated in the Ethereum context and illustrated with a case study.

**INDEX TERMS** Personally Identifiable Information, Blockchain, Ethereum, Security Management.

## I. INTRODUCTION

During these last decades, Identity Management Systems (IdM) [1] have been studied and developed for managing users' identifiers across systems. IdM have inherent flawns and complexities [2]. One of these flawns is to provide trust, and these systems must ensure the management of identities. Literature reveals several reserch works in this sense. For example, due to the increase of different interconnected systems some solutions are provided as Federated Identity Management Systems [3] or even for securing these federated systems [4].

Identity Management is a controversial concept [5], mainly because the different stakeholders have different views and requirements about how identites should be managed. This has resulted in quite a number of different approaches towards providing identity management such as [2] where authors provide guidelines about how to design a decentralized web identity management system. In fact, these guidelines include stakeholders' motivation as well as their capabilities. They also include the usability aspect as a key concept to achieve wide acceptance. The term Personally Identifiable information (PII) is defined in [6] as the information which can be used to distinguish or trace an individual's identity. In this IdM context, PII is a keystone concept [7] where stakeholders and systems should carefully manage the information they are processing. This information can be based on their name, social security number, biometric records, among others, taht can be combined with other personal or identifying information [7]. In fact, the loss of PII is a critical issue [8] not only from a legal or regulatory point of view, where systems must ensure the privacy of the data, but also from a personal point of view where users can lose the control of their data [11].

The concept of identity has become one of the research

topics in security and privacy areas [2] where the real identity of users must be preserved, and pseudo identities are created and used [12]. The identity is defined as the qualities, beliefs, personality, looks and/or expressions that make a person. Thus, an identity is made up of identifiers and attributes somehow linked to these identifiers. Since identifiers are often the only connector to an identity, revealing such connections is often target of attacks. There are methods not only for infering personal information from obfuscated data, but also for predicting the presence of private information such as in emails [13]. Therefore stakeholders must carefully use their data, even if they are not explicitely using their more sensitive information. Pseudo identities are then created and used setting hurdles and obstacles during the identification process, such as Personally Identifiable Honeytokens [14].

With the rise of Blockchain-based systems, identities are becoming more critical then before mainly due to its immutability property. In fact, many publicly accessible Blockchain networks like Ethereum rely on pseudonymization as a method for identifying a subject. Pseudonyms are often employed to maintain anonymity, but true anonymity requires unlinkability [15]. Without this property, any attacker can examine the messages sent by a specific pseudonym and learn new information about the holder of this pseudonym.

This is exactly what happens in Ethereum, where all transactions are publicly auditable. This has serious implications in the decentralized applications (DApps) which use Ethereum, because they generate a trace providing new information about the identity of the users. In other words, they break the non-binding requirement of true anonymity.

The European Union General Data Protection Regulation (GDPR) [9] and the California Consumer Privacy Act (CCPA) [10] are regulations in laws on data protection and privacy that address the processing, storage and transfer of personal data. These regulation types promote the personal privacy enhancement with terms like *right to be forgotten* and *right to delete personal information*, what collides with the immutability property obtained by using Blockchain-based solutions and responsible for ensuring that every data stored within the ledger will not be modified in the future.

The unlinkability of digital identities within Blockchain environments is the key research of this paper, while being compliant with regulations. The proposed method and tool provide a means to grant pseudo identities to users that are totally unrelated to their real identity. This method uses the benefits of key derivation systems to ensure a non-binding interaction between users and the information model associated with their identity. To demonstrate the feasibility of the anonymity of user actions and the unlinkability of identifiers the Ethereum context is great to demonstrate the method versus the current use of pseudonymous identities.

Remarkably, in the proposed Blockchain identity method, each user will have a set of pseudo identities that will remain secret for everyone else and all ledger operations will use these identities in a non-binding way. Giving three advantages: (i) is compliant with the *right to be forgotten*, (ii) only allows to trace user actions at application level (i.e., for DApps) and (iii) runs on top of the existing Ethereum networks (i.e., is compatible with Ethereum's design and philosophy).

The paper is structured as follows: Section II provides a background study on PII and Blockchain. Section III introduces and details our approach., Section IV describes the main software components for this architecture. Section V depicts how this solution is used in a case study. Section VI describes the results and provides a discussion about them. Finally, in Section VII the conclusions of the work are presented together with future research lines.

## II. BACKGROUND STUDY

### A. PERSONALLY IDENTIFIABLE INFORMATION

There are several definitions and references for what constitutes Personally Identifiable Information (PII). Organisations such as NIST provides their own PII definition [16], and they use to refer the OMB Memorandum M-07-1616 [7] which defines PII as the information that can be used to distinguish or trace an individual's identity, either alone or when combined with other personal or identifying information that is linked or linkable to a specific individual [17]. Other definitions are generalising the concept as any information (a) that identifies or can be used to identify, contact, or locate the person to whom such information pertains, (b) from which identification or contact information of an individual person can be derived, or (c) that allows linking particular personal characteristics or preferences to an identifiable person [18]. This PII concept dates back in a U.S. Privacy Act 1974 regulating the collection of personal information by government agencies [19]. It is widely known that the emergence of new powerful algorithms jeopardizes anonimization techniques [19].

In this sense, the protection of PII is envisaged as a hot topic and a cornerstone for any system involving personnal data. For example, differential privacy has been used for protecting PII in Critical Infrastructure Data [20]. This issue is not only related to critical systems, but also to every organisation collecting, processing, and transmiting customers' data or employees' data [21], because they are becoming attractive targets for cybercriminals.

Systems are becoming more complex and they are relying on privacy-enhancing technologies (PET), and personally identifiable data (PID) is at the core of any PET [18]. However, PID disclosure is a risk to be managed appropriately. In fact, in [22] authors are researching at the network traffic level, and they are proposing a Software Defined Networking (SDN) / Network Function Virtualization (NFV)-enabled architecture for improving the efficiency of leak detection systems. This is a similar approach to the one proposed for identifying PII in internet traffic [23].

Therefore, PII is a research topic *per se* where different methods are applied in order to minimize or to reduce the

disclosure of PID over the network.

### B. BLOCKCHAIN, PII AND REGULATIONS

Since the emergence of Blockchain, several applications have been reported and their benefits are clear. Some of these experiences are focused on the concept of digital identities such as [24] where authors are proposing some patterns in this context. From a practical point of view [25], the records stored in a ledger can't be altered, and therefore, Blockchain architects must identify which data is going to be used inside the records.

Bitcoin was originally designed for one only purpose, i.e., to create an unfeasible double-spending resistant electronic system built over an international peer-to-peer network where nodes running Bitcoin just relay and broadcast transactions to each other following several communication rules. In Bitcoin each user is identified by unique and personal ID composed as the 160-bit hash of the public portion of a public/private `ECDSA` keypair, and usually used encoded as `Base58` text string. More modern completely decentralized Blockchain networks follow a similar approach to identify users. Since Bitcoin addresses are generated from randomly seeded numbers, it is possible, although extremely unlikely, for two people to independently generate the same address.

More generally, the concept of Decentralized Identity can be redefined in terms of Asymmetric Cryptography, the Identity $I$ of user $U$ becomes a public-private key pair $(pubU, privU)$. The $pubU$ public key authenticates the client $C$ and links current operations to previously stored operations in the ledger These operations are linked using public identifiers known as addresses that are derived from $pubU$. The $pubU$, however, allows the user to send signed messages identified as $I$.

In the context of Blockchain the so called Self-Sovereign Identity (SSI) is gaining relevance and it is considered to be a "*killer application*" [24], especially in environments where data security and privacy are essentials. PII is an asset for many applications and since data is stored in decentralized manner these applications are required to implement multi-user system for access control to stored datasets [26] [27]. Other approaches lay on third party service accountability [47] or early stages of the SSI proofing concept [48] to protect attributes, but don't take into account that the identifier itself can mean a way of PII colliding with regulations.

One of the biggest challenges applying differential privacy [28] in this scenario is the identification of accurate PII parameters. As there is no predetermined rule to declare that the specific piece of information is counted as PII or not [28].

The extensive use of PII by social network applications (SNAs) users on the Internet has raised concerns for privacy advocates [18], and this is applicable to Blockchain [29]. For example, some Blockchain platforms have been modified as Blockchain-based transaction processing systems (TPS) [30] for the preservation of confidentiality. The use of personally identifiable attributes constituting a digital identity is an integral part of service transactions over networks and identity trust is being calling in question [31].

Other research works are focused on using Blockchain as IMS such as in [32] where a Blockchain-based Personally Identifiable Information Management System (BcPIIMS) is designed for PII management throughout organizations, in [49] where attribute managed user identities are certified and controled by authorities or in [50] for specific pre-created and permissioned groups of users.

Another Blockchain-based solution is the EIDM (Ethereum-based Identity Management) protocol [33]. This new protocol solves the problem of over-reliance on third parties in the existing identity management system solutions. The performance evaluation results also indicated that the new protocol demonstrates better practicability and flexibility [33].

However decentralization is not the solution for PII and there are some existing research works revealing these issues related to privacy preservation [34].

From a regulatory point of view, governments and agencies are stressing the PII concept, and depending on each case the use of PII is more restrictive than others. For example, the Health Insurance Portability and Accountability Act (HIPAA) [35] provides a set of rules for maintaining secure data storage, and to safely transmit patient PII. California Consumer Privacy Act (CCPA) [10] grants California consumers data privacy rights and control over their personal information, including the right to know, the right to delete, and the right to opt-out of the sale of personal information. Aligned to this idea, GDPR also protects any user of a system including the right to be forgotten. Our purpose is not to describe in detail all available regulations but to describe some of them and to highlight the fact that PII is the asset to be protected, and there is an international trend granting users and stakeholders the right to modify and to erase their PII. Therefore, the use of Blockchain-based solutions should be carefully implemented.

In order to find a balance between the benefits of using Blockchain, regulations and stakeholders' rights, some authors have proposed solutions such as [36] where authors proposed a LinkShare model using Blockchain to create a secure, centralized, immutable and trusted data privacy measurement framework.

### C. ETHEREUM ACCOUNTS

Ethereum (and other Ethereum-like clients) enhances the functionality offered by Bitcoin [37], providing the decentralized Ethereum Virtual Machine (EVM), that can execute scripts using a non-localized network of nodes. Those who are owners of an Ethereum account in this decentralized system can propose changes on the state ledger, signing transactions with their private key and making them auditable with their address. This need of using accounts is the base for the proposed method.

Ethereum based technologies rely on Elliptic Curve Digital Signature Algorithm (ECDSA). `ECDSA` is a pure public-key cryptography system that is mainly used to sign and verify messages. Ethereum based technologies implement `ECDSA` using `secp256k1` curve parameters. These private keys $sk$ and public keys $pk$ are created as part of an Ethereum account for every user $U$ that wants to have an identity $I$ on the network.

In addition, for each $I \in U$, the address $addr(U)$ is designed by the `Keccak-256` hash of $pk(U)$, taking the last 40 characters (20 bytes) and prefixing it with 0x. $addr(U)$ is the main identifier of $U$ when working in Ethereum and it is used for the next set of tasks:

- Contract installation.
- Contract calling and interaction.
- Identifier for incoming transactions as part of its payload.
- Identifier for account balance.
- Identifier for token trading and coins related managements.

## III. THE BURNABLE PSEUDO-IDENTITY METHOD

The concept of Pseudo-Identity has been used in several situations but recently has been used together with Blockchain approaches [38]. The concept of burnable stems from the implementation side where tokens are burnt.

The method presented in this paper embodies the creation, management and erasure of the burnable pseudo-identities for a user. This method is composed by a set of different steps, described by Figure 1:

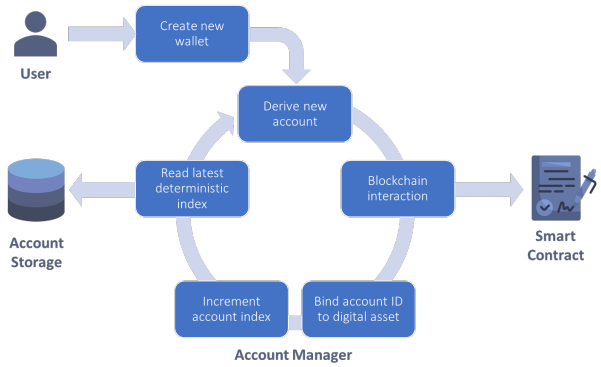


**FIGURE 1.** Burnable Pseudo-Identity account flow.

1) **Create new wallet**: The first action comprises the creation of a deterministic wallet $W$ that represents the burnable pseudo-identity. $W$ consists on a set of accounts $acct$ along the index space with size $goal$ made available $W = acct(i) \mid 0 > i > goal$. Encryption mechanisms ensure that $W_i$, and in consequence every $acct_i$, are unique. The access to the elements of $W$ is protected by a master key $mk$.
2) **Derive new account**: Each time the Blockchain network needs to be reached by the user $U$, a new $acc(U_i)$ is appended to $W$. $acc(U_i)$ is derived from $mk$ and the correspondant $i$ and it is composed by a public key $pk(U_i)$, a private key $sk(U_i)$ and an address $addr(U_i)$.
3) **Blockchain interaction**: The derived $acc(U_i)$ is responsible for signing the transaction, and $addr(U_i)$ will appear in the *from* field when calling a Smart Contract.
4) **Bind account to digital asset**: The transaction and $addr(U_i)$ are bound offchain, so the user can locally decide when to destroy the pair.
5) **Increment account index**: When the maximum number of usages for $addr(U_i)$ is reached, $i$ is incremented $(i \rightarrow i + 1)$ to allow the selection of the next account (enclosing $addr(U_{i+1})$). This responds to the rotation concept.
6) **Read latest deterministic index**: $addr(U_{i+1})$ is read and prepared for the next interaction.

Keeping low the number of usages for one account reduces the historical trace of transactions in the ledger. Altough the number of these usages can be customized, it is defaulted to 1. This way, there is only one pair binding between each account identifier and the user digital assets, and there is no historical trace of transactions.

### A. IDENTITY CREATION

Our approach defines the burnable pseudo-identity which is a Web3js based implementation for representing digital identities. The creation detailed in Figure 2 involves an initial seed, from a secure entropy source, and a user secret. The secret is used during the identity storage to symmetrically encrypt it.

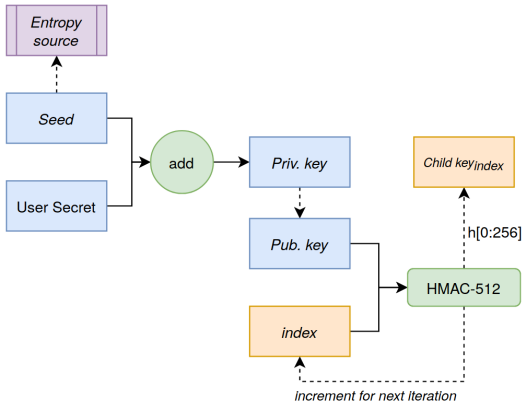


**FIGURE 2.** Identifiers creation process.

The process of creating an account is based on the Ethereum HD standards BIP 32 [39] and BIP 44 [40]. Following the BIP 39, we also provide an account recovery mechanism for users based on mnemonics. The real complexity of burnable pseudo-identities lies in how identities are used to satisfy previous legal and compliance requirements assuring that they usage is forbidden after their lifetime period or upon explicit user requirement.

### B. IDENTITY ERASURE

The identity erasure can be requested at any time to the system where the Burnable Pseudo-Identity Method is used. The request may come directly from the user or caused by the end of a service. As established by the GDPR, art. 17, *"The data subject shall have the right to obtain from the controller the erasure of personal data concerning him or her"* [9]. When this action takes place, Data Controllers services or DApps must "forget" any data related to that specific user. As Blockchain keeps an immutable, traceable, forever growing record of the actions taken place in the network, a special effort has been made to achieve this need.

Technical research has been done to find the best way to unlink PII data from the system without breaking the block record and without rejecting the Blockchain original philosophy, that would have ended in a misuse of the technology. The solution has embraced the fact that if the system cannot access the data, the data becomes atomic, untraceable and anonymized.

When a Data Subject sends a message $m_i$ containing a piece of atomic information, the Data Processor can verify that $m_i$ has been sent by $addr(U_i)$, but he will not be able to link $m_i$ to any previous messages $(m_j \mid 0j < i)$, since they have been created by independent accounts $(addr(U_k) \mid k = j)$. This assertion is also true once the relation between them is over, because the observer can't infer new information about the sender $(m_j \mid i < j)$.

Summarizing, the dissociation between the atomic data and the Data Subject leads to a full anonymization of the data. The identifier rotation is a key point of the Burnable Pseudo-Identity Method, constantly changing in a circular array fashion.

### C. ROTATION AND STORAGE OF IDENTIFIERS

At some extent, the burnable pseudo-identities are defined as a circular array of size n – where n is the maximum number of identities to be created and managed simultaneously – creating an endless pool of burnable pseudo-identities (see Figure 3). Each pool structure is used for each DApp or Data processor service the user wants to interact with, so it provides full anonymization and privacy against the ledger interaction, ledger monitoring, and simultaneous DApp usage. When the pseudo-identities are burnt by the user, the relationships made on the ledger between a digital asset and a public address become irresoluble. At that point, there is no feasible way to lookup nor recover the original author or entity behind a transaction.

Since burnable pseudo-identities are compatible with Ethereum identifiers, they can also be stored in an *Ethereum V3 KeyStore*; an encrypted way of storing private keys used for signing transactions. If users lose this file, users lose access to their unique private key and to the ability to sign and execute transactions. This process is unrecoverable unless proper recovery mechanisms are designed.

Our current key storage mechanism requires to encode information using at least `AES-128-CTR` crypto protocol

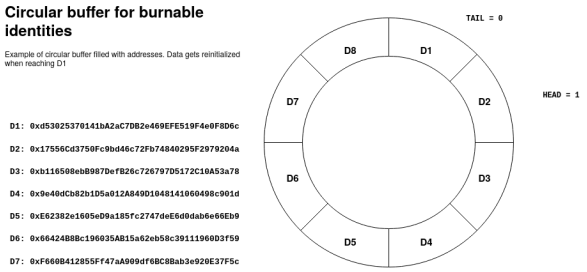


**FIGURE 3.** Circular pool of accounts with $N = 8$

in order to store data safely in filesystem. By default, and following conventions, keystore filenames are `128-bit UUID` given to the secret key and saved as uuid.json. These files have an associated password chosen by the User. To derive a given uuid.json file's secret key, first we derive the file's encryption key; this is done through taking the file's password and passing it through a key derivation function as described by the kdf value. KDF-dependent static and dynamic parameters to the `KDF` function are described in kdfparams value. We preset `KDF` (key derivation function) to `PBKDF2`, being `PBKDF2` kdfparams as follows:

1) `prf`: `hmac-sha256`.
2) `c`: number of iterations to be made in `KDF` routine.
3) `salt`: salt passed to `PBKDF` algorithm.
4) `dklen`: length for the derived key. Must be bigger than `32` bytes.

Once the file's key has been derived, it should be verified through the derivation of the `MAC`. The `MAC` should be calculated as the `SHA3` (`keccak-256`) hash of the byte array formed as the concatenations of the second-leftmost `16` bytes of the derived key with the ciphertext key's contents. Finally, User Identities are stored.

## IV. TOOLS AND SOFTWARE COMPONENTS

The user privacy is ensured by the formulation of involved components when interacting with Ethereum through DApps and Data Processor services. The method works both at client-side (for DApps running in browsers) and server-side (for libraries running in host servers).

To achieve that objective, a design based on JavaScript (abbreviated JS) and WebAssembly modules (abbreviated Wasm) is suggested. Nevertheless, the method outlined in this paper establishes the objective and the communication flow of the components but is independent of the internal design and implementation of each component.

The relevant components of the proposed method are Keygen, Account Evaluator, Account Manager and Web3 TX Manager. Figure 4 details the relationship among them and with other parts in the system. The Burnable Pseudo-Identity Method is a blackbox allowing the management of privacy-aware identities for user-centric decentralized applications.

The users will interact directly with the Burnable Pseudo-Identitiy Method implementation to manage their accounts

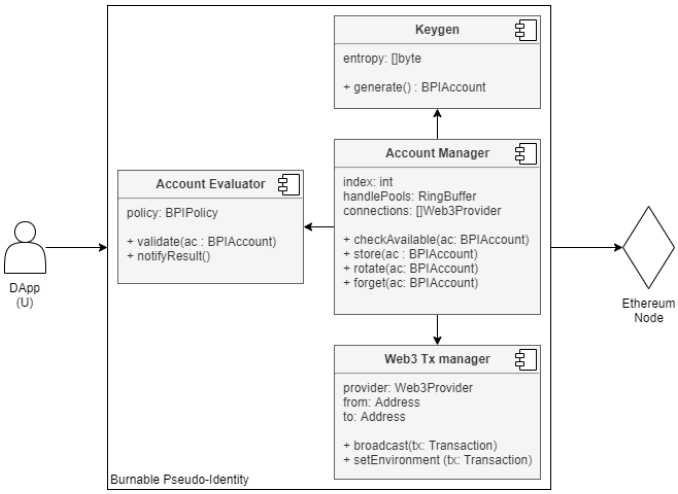


**FIGURE 4.** Software components involved on the Burnable Pseudo-Identity management .

and interact with the node of an Ethereum network.

The Account Manager is the main module orchestrating the flow.

1) It manages the accounts created by the Keygen, indexing, listing, rotating and storing them.
2) It gets the current account from the Account Evaluator.
3) It requests Web3 Tx Manager to construct the transaction packet, sign it with a proper burnable pseudo-identity, and broadcast it to the network.

### A. ACCOUNT MANAGER

The Account Manager ensures a proper off chain user-centric creation and operation for the set of identities $I$ belonging to a user $U$. The objective is to relate the accounts owned by the user, noted as $U_i$ , being the index $i$, to those used as part of previously committed transactions, satisfying (1) and (2).

$$\sum_{i=1}^{n} U_i \equiv CKDF(i, U_{pk}, seed) \tag{1}$$

$$I \in \sum_{i=1}^{n} U_i \tag{2}$$

The account creation is ruled by the defined "*rotation and storage of identifiers*" mechanism proposed in this manuscript.

The *store* function allows to store *M* in the account pool *I* of the user U to make it recoverable as described before in this manuscript.

The *forget* function performs the procedures for safe data deletion. As the set of identifiers *I* is kept locally, only information detached from the user *U* remains on the network when *I* is removed. This function addresses the 'right-to-be-forgotten', making the method compliant with current regulations.

The *rotate* function performs an on-demand rotation of *I* and updates the contents of the *handlePools* with new identities for *U*. To enhance the trust model that avoids arbitrary rotation requests, this function always requires the direct interaction from *U*, providing the wallet authentication passphrase.

The *checkAvailable* function returns the first available burnable pseudo-identity. To do it, the Account Manager checks its local state and the connections. Checking the availability ensures that the implementations based on this Method will follow the Security by Design approach. If no burnable pseudo-identity is available, *checkAvailable* will call rotate function.

### B. KEYGEN

Keygen is the managing component in charge of gathering secure entropy values and creating strong cryptographic accounts for end users as described by Algorithm 1.

The *generate* function allows to create new accounts based on HD Wallets for the user. The process includes the gathering of a secure entropy origin, formed by a []byte slice filled with noise from a customized random cryptographic secure source. With the entropy input, it creates the accounts that are compliant with the EC `secp256k1` Ethereum standard [41]. The function adds the required metadata *M* to the account, where the minimum set of *M* is: (i) *key seed* length, (ii) *r* (the public key recovery parameter), (iii) *i* (the current heuristic index) and (iv) *kdfparams* (explained in "rotation and storage of identifiers"). Each account gets encrypted (passphrase protected) to be stored while not being used and it will be recoverable by the user since it belongs the pool of burnable pseudo-identities owned internally.

**Algorithm 1:** Account creation process

**Result:** W
let ent_size ← x;
let entropy[ent_size];
let W, M;
**while** $k < ent\_size$ **do**
  entropy[k] ← rnd();

**end while**
$W \leftarrow new\ secp256k1(entropy)$;
$M \leftarrow \{seed\_length, r, i, kdfparams\}$;
$W \leftarrow add(M)$;
$W \leftarrow encrypt(W, passphrase)$;
return: W;

### C. ACCOUNT EVALUATOR

Account Evaluator is the component that gets and validates the account to be used. As the validation response, a subsequent request is sent to the Account Manager for updating the pool of available addresses for the given user.

The *validate* function performs all the tasks related to the account address validation. It first validates whether the given account address was created by the corresponding Account

Manager and that this address is recognized in the user local information. After this initial validation, the Account Evaluator validates whether the created burnable pseudo-identity is not breaking the number of uses constraint (default to 1) per account and that it is valid from a privacy point of view.

The *notifyResult* function notifies the result obtained from the validation process to the Account Manager, indicating whether it should create a new burnable pseudo-identity or use any of the pooled ones. If the validate function rejects the use of a requested account during the process, it will be notified to the Account Manager. Then, the next available account from I will be forced to be used.

### D. WEB 3 TX MANAGER

The Web3 Tx Manager is a transaction manager based on a web3 implementation. This module is a medium to process the details of a raw transaction, marshal them and broadcast the ABI encoded Json-RPC messages to a selected provider. The provider can be defined as any valid Ethereum protocol compatible peer running an RPC listener on its backend.

Once the account has been selected and validated by the Account Evaluator for the following transaction, the *setEnvironment* function configures the Web3 TX Manager using those transaction parameters. The field to as the destination address, the field *from* to sign the transaction, and the message, serialized as ABI encoded payload.

The *broadcast* function connects with the customized provider, fixed in the field *provider*, and delivers the message.

## V. CASE STUDY: USE OF THE BURNABLE PSEUDO-IDENTITY METHOD FOR PLATFORMS NOT MANAGED BY USERS

The creation of a burnable pseudo-identity is always triggered by a conscious action of the wallet owner. It is highly recommended that the user accounts are managed locally, and that a user controlled DApp oversees using the account and building the transaction.

When the use case requires to delegate the user actions in a middleware entity (e.g. a platform provider), the level of security and privacy offered must remain the same.

In such scenarios, the provider must create, manage and store the accounts with authorization of the users. The random seed (as entropy source) is created by the provider and offered as a service, but the secret for symmetric identity encryption is only known by the user and the identity remains hidden and securely protected while it's not being used.

All these actions take place to ensure the compliance with ethic requirements. One of them is the prohibition of using an identifier after its lifetime period, so the data must be created in a way that it can be unlinked if needed. This is the purpose of the Burnable Pseudo-Identity Method.

The provider entity must ensure the authentication flow as shown in Figure 5

Given a city council public service as Authoritative Server *A* and a citizen as User *U*, it is assumed a scenario where

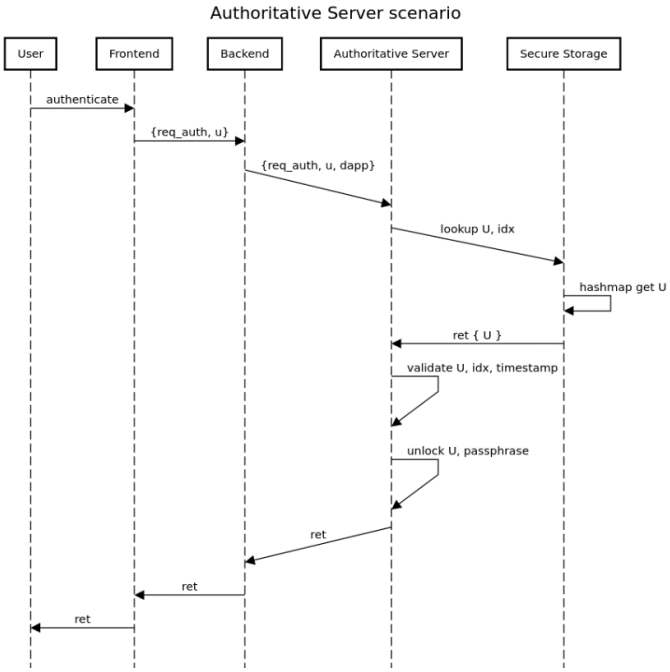


**FIGURE 5.** Non-user managed authentication flow.

A delegates its permission management system in a network based on Ethereum technology. The observation of transactions in this permission network would enable a third party to associate transactions requested from the citizen's account to be executed by the smart contract. For this reason, it is possible to infer the immutable profile of the citizen, limiting compliance with regulations.

In the proposed use case, *U* interacts with the system through a frontend provided by A. This frontend is connected to a backend *A* in charge of providing the service. Now, let Burnable Pseudo-Identity Method be in charge of managing Ethereum accounts for *U* when using *A*. The authentication in *A* comprises the finding of the user wallet, uniquely identified and secured in a Secure Storage module. Once the wallet is recovered and verified, it must be decrypted and unlocked to allow the account usage.

When the city council is using this approach to manage the citizen permissions, this will allow to keep the citizen identity unknown to any observer. When *U* executes the *right to be forgotten* the Secure Storage can safe-delete the HD wallet belonging to *U*, and the only information left from *U* will be isolated permissions from different accounts.

## VI. RESULTS AND DISCUSSION

### A. PROPOSED ENHANCEMENTS IN IDENTITY MANAGEMENT

The method described in this paper improves the Identity management offered by traditional of non-permissioned Blockchains in four aspects.

1) **User centric traceability**. Despite of using a Blockchain system, proposed *burnable pseudo-*

*identities* are only traceable from user point of view due to the irreversibility of hierarchical deterministic algorithms. Moreover, any third party that reads ledger transactions (e.g., Ledger Explorers) will not be able to correlate, link or build user centric analytics. This adds an extra layer of privacy without modifying Ethereum protocol nor codebase.

2) **Right to be forgotten**. When a Data Subject decides to stop using a DApp or Data Processor service the provider is required to erase all the PII from its platform to ensure the compliance with legal regulations like GDPR, CCPA or WPA. Since modifying data stored in a Blockchain platform is not an option due to its immutability, it is mandatory to study and design a way to unlink the PII in a way that it is fully unrecoverable, no matter the attack vector.
   Our solution achieves this providing a circular array for a pool of burnable pseudo-identities for each user and forgetting them upon request. Interactions made at a Smart Contract level to update the ledger will still create a history over time but binding two observed interactions from the same user will not be possible only observing this history. Therefore, no one will be able to link data stored in the ledger to a user once these pseudo-identities are burnt.
3) **Unlinkability**. The Ethereum identities used in a circular array have the irreversible property of `HMAC-SHA512` algorithm and Child Key Derivations (CKDF) functions for both public and private keys. This allows our method to create new child keys from parent keys but not otherwise. Breaking a reversibility will constitute a 'seconds preimage' attack [42] on `PBDKF2-SHA-512`.[43]
4) **Privacy**. Only users with proper identities will be able to recover stored data. Currently, stored information remains public for everyone in public Blockchains due to the design of these networks.

It is strongly recommended to implement the Burnable Pseudo-Identity Method at client-side when possible. The use of this recommendation avoids server-side related security breaches, like account leakage. However, in scenarios where the workflow involves additional external entities, it is the duty of these to guarantee a proper user data security and privacy established by the security-by-design patterns.

### B. BRUTE FORCING PROTECTION

The benefits achieved with the proposed method rely on the practical inability to generate the exact same seed used to generate all the identities for a user. This section gives estimations on the strength of this claim.

Our solution follows the BIP 39 specification [44] to generate the seed needed to start creating new identities for each user with the BIP 32 method. BIP 39 deterministically generates seeds based on a mnemonic which is intended to be more memorable and readable for a user than random bits.

BIP 39 states that initial entropy can only come in a few sizes: multiples of 32 bits, between *128* and *256*. Together with a checksum (CS represents its length), this entropy is encoded in a combination of *2048* memorable words. From Table 1, it can be compared the mnemonic size needed for all the possible sizes, together with the search space that an attacker would need to traverse to look for any given entropy. The search space varies from $5.44 \cdot 10^{39}$ in the less secure scenario (using a 12 word mnemonic) to $2,96 \cdot 10^{79}$ in the most secure one (24-word mnemonic).

| mnemonic size | bits size | *CS* length | entropy | search space |
|---|---|---|---|---|
| 12 | 132 | 4 | 128 | $5.44 \cdot 10^{39}$ |
| 15 | 165 | 5 | 160 | $4.67 \cdot 10^{49}$ |
| 18 | 168 | 6 | 192 | $4.01 \cdot 10^{59}$ |
| 21 | 231 | 7 | 224 | $3.45 \cdot 10^{69}$ |
| 24 | 264 | 8 | 256 | $2,96 \cdot 10^{79}$ |

**TABLE 1.** Mnemonic size reference data.

However, the seed is not only derived from this entropy but also from a passphrase provided by the user. The list of memorable words and how they combine with each other change from one client to another. In consequence, the security and reliability of this implementation don't vary from other Ethereum BIP standards.

Considering the worst case (not providing a passphrase or setting it as an empty string) and that the wordlist is known beforehand, the Figure 6 shows the complexity required to compute all the possible seeds in different search spaces with commodity hardware, being tested in a workstation with an Intel I5 and 8Gb RAM. It can be noted that the time needed to brute-force it increase exponentially compared to the search space length.

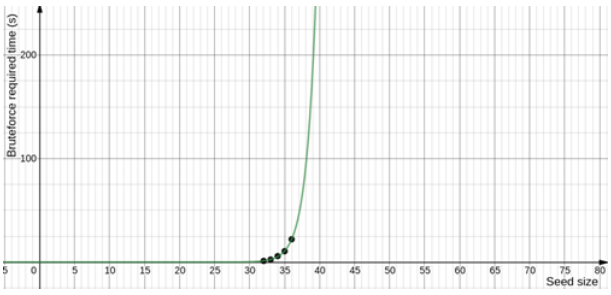


**FIGURE 6.** An estimation of complexity evolution to compute all possible seed sizes.

The computational cost required to simply brute force a search space of N bits is already unfeasible. Since `PBKDF2` can be implemented using very little RAM, it is known for not being resistant to ASIC and GPU attacks [45].

Therefore, the combination of specialized hardware and distributed HPC brute-force algorithms, could considerably decrease the time needed to search any space [46].

$$y_1 \sim ax + b \,|\, a = 1.7835 \cdot 10^{10} \wedge b = 2.03295 \quad (3)$$

Despite this, the complexity to bruteforce only one account (see Equation (3)) with the current hardware and recovering the original private key makes these attacks unfeasible due to the amount of time and resources required to success, and even in that case the Burnable Pseudo-Identity Method would only uveil a small portion of data compared to other approaches.

### C. IMPACT ON PERFORMANCE

To summarize, a scenario has been designed to run a test that allows to compare the performance impact on DApp operating interactions. In the test, Alice creates new standard Ethereum accounts each time she wants to execute a Blockchain transaction (algorithm implementation described in Algorithm 2).

**Algorithm 2:** Standard Ethereum account creation process.

```
Result: W
let goal ← x;
let W[goal];
while i < goal do
    s ← seed;
    addr ← create_address(s);
    sk ← create_private_key(s);
    W[i] ← addr, sk;
end while
return: W;
```

On the other hand, Bob uses the burnable pseudo-identities proposed in this manuscript (algorithm implementation described in Algorithm 3). Both approaches achieve the same anonymity, ignoring that Alice will need to manage all her accounts independently.

**Algorithm 3:** `BPI` account creation process.

```
Result: W
let goal ← x;
let W[goal];
s ← seed;
r ← create_mnemonic(s);
i ← 0;
path_0 ← create_path("m/44'/60'/0'/0/0") ;
while i < goal do
    W[i] ← addr, sk ← derive(path_0, i) i ← i + 1 ;
end while
return: ← W[i]|i ≥ 0 ∧ i < max;
```

The results of both algorithms show that even when the generation of the set of burnable pseudo-identities has a warmup stage, where accounts are derived from the initial seed $s$, a maximum amount goal and a derivation path $path_0$, the overall result at runtime is much faster than on-demand creation of standard Ethereum accounts. The results of this experiment are described in Table 2, where it is evidenced that the proposed method improves the base implementation, allowing smoother workflows and less delays between the account request and the generation steps.

| *goal* parameter value | `BPI` method | `Standard` method |
|---|---|---|
| 1 | 13.125 $\mu$s | 88.92 $\mu$s |
| 100 | 102.025 $\mu$s | 6.285669 ms |
| 500 | 615.405 $\mu$s | 33.427533 ms |
| 1000 | 4.977616 ms | 67.209922 ms |

**TABLE 2.** Model time comparison to create $goal$ accounts.

Research has been conducted [51] [52] [53] to evaluate the performance on Ethereum HD wallet implementations through experiments. A `go` reference implementation [51] has been selected to be compared with the results of the proposed method. This implementation allows to make simulations with a reference using the same programming language. The purpose of the experiment is to challenge the previous studies with this one and demonstrate that the anonymization of the pseudonym environments has no negative impact on the performance. The experiment is composed by several simulations and it has been focused on evaluating the performance of creating an increasing number of accounts ($goal$) in each simulation round. The approach has been to select different $goal$ parameter values and measure three parameters: (i) being $t$ the required time to complete, (ii) $p$ the impact on CPU and (iii) $m$ the memory usage. Seen in Algorithm 4, it helps to compare the performance of the presented method (see Algorithm 3) and the reference method (see Algorithm 2). On each round of the Algorithm 4 , the $goal$ parameter is updated with a $goal_i = 2^i \cdot 100 | i >= 0$ pattern to increase the difficulty of the experiment. To reduce the noise while increasing the statistical results, the `go test tool` used for the simulations has been tweaked with the flags `-benchtime=2s` and `-count=10`. With the aim of making this experiment replicable, it has been performed on a Linux Workstation with an Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz and 16 GB RAM.

Table 3 shows the reference HD Wallet implementation simulation results, while Table 4 shows the results of the simulation for the proposed implementation. A comparison made in the Table 5 reveals that the proposed method is rewarded with performance improvements over the reference implementation, resulting in an average of 51.24% faster in CPU and with 50.04% less memory usage. All the tables are presented in compliance with the Go Benchmark Data Format [54].

The experiments have also disclosed that the improvement ratio is limited by hardware due to the cryptographic nature of the process. The limitation arises when random seeds are created under heavy cryptographic operations. Figure 7 shows this limit for `ScalarBaseMult` over a Kobliz curve. In the figure, `Mul2` operation takes `37,62%` over the total

**Algorithm 4:** Performance comparison model

```
Result: W
let rounds ← x ;
let count ← y ;
let benchtime ← z ;
let W[rounds] ;
i ← 0 ;
while i < rounds + 1 do
    goal ← 2 ^ i · 100 t← 0 ;
    p ← 0 ;
    m ← 0 ;
    j ← 0 ;
    let Wcount[count] ;
    while j < count do
        let Wtemp[] ;
        k ← 0 ;
        l ← 0 ;
        while k < benchtime do
            t, p, m, k ←
              eval(create_bpi_accounts(goal)) ;
            Wtemp[l] ← t, p, m ;
            l ← l + 1 ;
        end while
        Wcount[j] ← avg(Wtemp) ;
        j ← j + 1 ;
    end while
    W[i] ← avg(Wcount) ;
    i ← i + 1 ;
end while
return: ← W;
```

| *goal* parameter | time/op | bytes/op | alloc/op |
|---|---|---|---|
| 100 | 150ms | 1.97MB | 25.9k |
| 200 | 307ms | 3.94MB | 51.9k |
| 400 | 577ms | 7.89MB | 104k |
| 800 | 1.19s | 15.8MB | 209k |
| 1600 | 2.31s | 31.6MB | 417k |
| 3200 | 4.68s | 63.1MB | 835k |
| 6400 | 9.30s | 126MB | 1.67M |
| 12800 | 18.5s | 252MB | 3.34M |
| 25600 | 36.7s | 504MB | 6.68M |
| 51200 | 73.3s | 1.01GB | 13.4M |

**TABLE 3.** Reference measures for Ethereum HD Wallet implementation.

execution time and `SquareVal` operation takes `33,24%` over the total execution time.

## VII. CONCLUSION

In this paper, we have presented a novel manner to interact with existing Blockchain networks (Ethereum) without generating a trace which can be linked back to an identity. By using the Burnable Pseudo-Identity Method, we allow the user to sign his/her transactions with different Ethereum identities which cannot be related to each other and which

| *goal* parameter | time/op | bytes/op | alloc/op |
|---|---|---|---|
| 100 | 72ms | 0.99MB | 11.4k |
| 200 | 140ms | 1.97MB | 22.7k |
| 400 | 290ms | 3.92MB | 45k |
| 800 | 0.58s | 7.8MB | 91k |
| 1600 | 1.16s | 15.8MB | 182k |
| 3200 | 2.30s | 31.6MB | 363k |
| 6400 | 4.57s | 63MB | 0.73M |
| 12800 | 9.1s | 126MB | 1.45M |
| 25600 | 18.0s | 252MB | 2.91M |
| 51200 | 35.7s | 0.50GB | 5.8M |

**TABLE 4.** `BPI` Wallet algorithm implementation measures, CPU and memory metrics.

| *goal* parameter | Δ time/op | Δ bytes/op | Δ alloc/op |
|---|---|---|---|
| 100 | 51.79% | 49.92% | 56.14% |
| 200 | 54.28% | 50.00% | 56.25% |
| 400 | 49.80% | 50.27% | 56.36% |
| 800 | 51.64% | 50.36% | 56.44% |
| 1600 | 49.89% | 50.01% | 56.47% |
| 3200 | 50.98% | 49.98% | 56.49% |
| 6400 | 50.84% | 49.98% | 56.50% |
| 12800 | 50.96% | 49.98% | 56.51% |
| 25600 | 51.01% | 49.97% | 56.51% |
| 51200 | 51.27% | 49.97% | 56.51% |
| *avg* | 51.24% | 50.04% | 56% |

**TABLE 5.** Execution time, CPU and memory usage comparison results between `BPI` Wallet implementation and reference HD Wallet implementation.

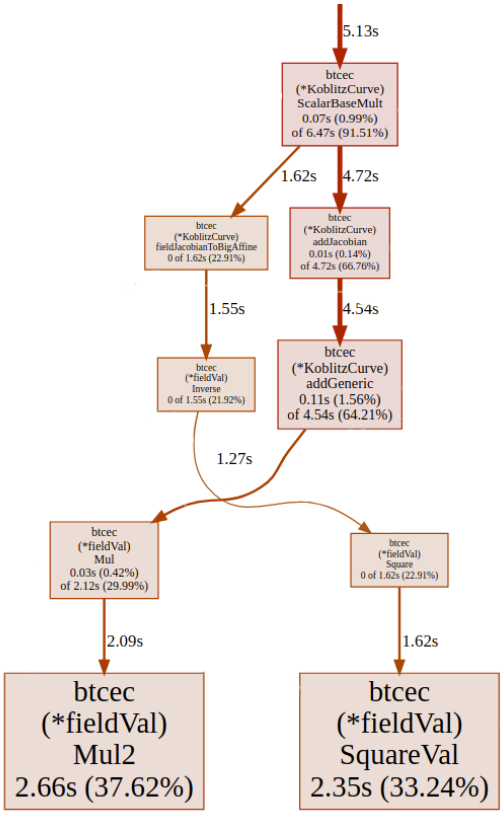


**FIGURE 7.** Kobliz curve scalar multiplication elapsed time for functions `Mul2` and `SquareVal`

can be discarded afterwards to achieve the *right to be forgotten*.

To the best of our knowledge, we have presented the first solution which considers privacy regulations to make user operations in Blockchain untraceable without interfering with the normal operation of the underlying network nor modifying its codebase. In contrast, this ability to keep the traceability of its transactions is only granted to the user at the application level. This privacy improvement is expected to increase users' confidence in Blockchain ecosystems and to contribute to the adoption of Blockchain technology in industrial use cases were the usage of personal data was preventing them to use Blockchain to date.

The next steps of this work will focus on (1) further generalizing the solution to make it agnostic to the underlying Blockchain framework, (2) fostering its adoption by seamlessly integrating it as a module for reference Web3 implementations or submitting it as an Ethereum Improvement Proposal[1] and (3) extending the current proposal to allow users to control their identity rotation by means of an external application.

## ACKNOWLEDGMENT

The authors acknowledge all the members of the POSEID-on Consortium for their valuable help.

## REFERENCES

[1] K. Tracy, "Identity management systems," IEEE Potentials, vol. 27, no. 6, pp. 34–37, 2008, doi: 10.1109/MPOT.2008.929295.

[2] R. Dhamija and L. Dusseault, "The Seven Flaws of Identity Management: Usability and Security Challenges," IEEE Security Privacy Magazine, vol. 6, no. 2, pp. 24–29, Mar. 2008, doi: 10.1109/MSP.2008.49.

[3] A. K. Sharma and C. S. Lamba, "Survey on Federated Identity Management Systems," in Recent Trends in Networks and Communications, vol. 90, N. Meghanathan, S. Boumerdassi, N. Chaki, and D. Nagamalai, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 509–517.

[4] U. Habiba, R. Masood, and M. A. Shibli, "Secure Identity Management System for Federated Cloud Environment," in Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, vol. 569, R. Lee, Ed. Cham: Springer International Publishing, 2015, pp. 17–33.

[5] G. Alpár, J.-H. Hoepman, and J. Siljee, "The Identity CrisisSecurity, Privacy and Usability Issues in Identity Management," Jan. 02, 2011. http://www.cs.ru.nl/J.H.Hoepman/publications/identity-crisis.pdf (accessed May 07, 2020).

[6] R. Alhajj and J. Rokne, Eds., "Personally Identifiable Information," in Encyclopedia of Social Network Analysis and Mining, New York, NY: Springer New York, 2018, pp. 1790–1790.

[7] Executive Office of the President- White House, "Safeguarding Against and Responding to the Breach of Personally Identifiable Information," May 2007. https://www.whitehouse.gov/sites/whitehouse.gov/files/omb/memoranda/2007/m07-16.pdf (accessed May 07, 2020).

[8] L. Wilbanks, "The Impact of Personally Identifiable Information," IT Professional, vol. 9, no. 4, pp. 62–64, Jul. 2007, doi: 10.1109/MITP.2007.77.

[9] THE EUROPEAN PARLIAMENT AND OF THE COUNCIL, "Directive 95/46/EC (General Data Protection Regulation)." Official Journal of the European Union, Apr. 27, 2016, Accessed: Jun. 25, 2019. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[10] the Office of the California Attorney General, "California Consumer Privacy Act of 2018." http://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?lawCode=CIV&division=3.&title=1.81.5.&part=4.&chapter=&article=.

[11] J. Isaak and M. J. Hanna, "User Data Privacy: Facebook, Cambridge Analytica, and Privacy Protection," Computer, vol. 51, no. 8, pp. 56–59, Aug. 2018, doi: 10.1109/MC.2018.3191268.

[12] J. Breebaart, C. Busch, J. Grave, and E. Kindt, "A Reference Architecture for Biometric Template Protection based on Pseudo Identities," 2008.

[13] L. Geng, L. Korba, X. Wang, Y. Wang, H. Liu, and Y. You, "Using Data Mining Methods to Predict Personally Identifiable Information in Emails," in Advanced Data Mining and Applications, vol. 5139, C. Tang, C. X. Ling, X. Zhou, N. J. Cercone, and X. Li, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 272–281.

[14] J. White, "Creating Personally Identifiable Honeytokens," in Innovations and Advances in Computer Sciences and Engineering, T. Sobh, Ed. Dordrecht: Springer Netherlands, 2010, pp. 227–232.

[15] Y. Liu, D. He, M. S. Obaidat, N. Kumar, M. K. Khan, and K.-K. Raymond Choo, "Blockchain-based identity management systems: A review," Journal of Network and Computer Applications, vol. 166, p. 102731, Sep. 2020, doi: 10.1016/j.jnca.2020.102731.

[16] National Institute of Standards and Technology (NIST), "Guide to Protecting the Confidentiality of Personally Identifiable Information (PII)." https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf.

[17] A. Wheeler and M. Winburn, "Application Data in the Cloud," in Cloud Storage Security, Elsevier, 2015, pp. 23–55.

[18] S. Weiss, "Privacy threat model for data portability in social network applications," International Journal of Information Management, vol. 29, no. 4, pp. 249–254, Aug. 2009, doi: 10.1016/j.ijinfomgt.2009.03.007.

[19] A. Narayanan and V. Shmatikov, "Myths and fallacies of 'Personally Identifiable Information,'" Communications of the ACM, vol. 53, no. 6, pp. 24–26, Jun. 2010, doi: 10.1145/1743546.1743558.

[20] A. Alnemari, R. K. Raj, C. J. Romanowski, and S. Mishra, "Protecting Personally Identifiable Information (PII) in Critical Infrastructure Data Using Differential Privacy," in 2019 IEEE International Symposium on Technologies for Homeland Security (HST), Woburn, MA, USA, Nov. 2019, pp. 1–6, doi: 10.1109/HST47167.2019.9032942.

[21] A. J. Burns and E. Johnson, "The Evolving Cyberthreat to Privacy," IT Professional, vol. 20, no. 3, pp. 64–72, May 2018, doi: 10.1109/MITP.2018.032501749.

[22] S. J. Y. Go, R. Guinto, C. A. M. Festin, I. Austria, R. Ocampo, and W. M. Tan, "An SDN/NFV-Enabled Architecture for Detecting Personally Identifiable Information Leaks on Network Traffic," in 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), Zagreb, Croatia, Jul. 2019, pp. 306–311, doi: 10.1109/ICUFN.2019.8806077.

[23] Y. Liu, H. H. Song, I. Bermudez, A. Mislove, M. Baldi, and A. Tongaonkar, "Identifying Personal Information in Internet Traffic," in Proceedings of the 2015 ACM on Conference on Online Social Networks - COSN '15, Palo Alto, California, USA, 2015, pp. 59–70, doi: 10.1145/2817946.2817947.

[24] Y. Liu, Q. Lu, H.-Y. Paik, X. Xu, S. Chen, and L. Zhu, "Design-Pattern-as-a-Service for Blockchain-Based Self-Sovereign Identity," IEEE Software, pp. 0–0, 2020, doi: 10.1109/MS.2020.2992783.

[25] C. Ebert, P. Louridas, T. M. Fernandez-Carames, and P. Fraga-Lamas, "Blockchain Technologies in Practice," IEEE Software, vol. 37, no. 4, pp. 17–25, Jul. 2020, doi: 10.1109/MS.2020.2986253.

[26] I. Sukhodolskiy and S. Zapechnikov, "A blockchain-based access control system for cloud storage," in 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), Moscow, Jan. 2018, pp. 1575–1578, doi: 10.1109/EIConRus.2018.8317400.

[27] S. Aggarwal, R. Chaudhary, G. S. Aujla, N. Kumar, K.-K. R. Choo, and A. Y. Zomaya, "Blockchain for smart communities: Applications, challenges and opportunities," Journal of Network and Computer Applications, vol. 144, pp. 13–48, Oct. 2019, doi: 10.1016/j.jnca.2019.06.018.

[28] M. U. Hassan, M. H. Rehmani, and J. Chen, "Differential privacy in blockchain technology: A futuristic approach," Journal of Parallel and Distributed Computing, Jun. 2020, doi: 10.1016/j.jpdc.2020.06.003.

[29] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, "A survey on privacy protection in blockchain system," Journal of Network and Computer Applications, vol. 126, pp. 45–58, Jan. 2019, doi: 10.1016/j.jnca.2018.10.020.

[30] Y. Wang and A. Kogan, "Designing confidentiality-preserving Blockchain-based transaction processing systems," International Journal of Accounting Information Systems, vol. 30, pp. 1–18, Sep. 2018, doi: 10.1016/j.accinf.2018.06.001.

[31] P. Pacyna, A. Rutkowski, A. Sarma, and K. Takahashi, "Trusted Identity for All: Toward Interoperable Trusted Identity Management

[1] https://eips.ethereum.org/

Systems," Computer, vol. 42, no. 5, pp. 30–32, May 2009, doi: 10.1109/MC.2009.168.

[32] N. Al-Zaben, M. M. Hassan Onik, J. Yang, N.-Y. Lee, and C.-S. Kim, "General Data Protection Regulation Complied Blockchain Architecture for Personally Identifiable Information Management," in 2018 International Conference on Computing, Electronics Communications Engineering (iCCECE), Southend, United Kingdom, Aug. 2018, pp. 77–82, doi: 10.1109/iCCECOME.2018.8658586.

[33] S. Wang, R. Pei, and Y. Zhang, "EIDM: A Ethereum-Based Cloud User Identity Management Protocol," IEEE Access, vol. 7, pp. 115281–115291, 2019, doi: 10.1109/ACCESS.2019.2933989.

[34] L. Bahri, B. Carminati, and E. Ferrari, "Decentralized privacy preserving services for Online Social Networks," Online Social Networks and Media, vol. 6, pp. 18–25, Jun. 2018, doi: 10.1016/j.osnem.2018.02.001.

[35] "Health Insurance Portability and Accountability Act of 1996 (HIPAA)." https://www.cdc.gov/phlp/publications/topic/hipaa.html (accessed May 07, 2020).

[36] A. Banerjee and K. P. Joshi, "Link before you share: Managing privacy policies through blockchain," in 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, Dec. 2017, pp. 4438–4447, doi: 10.1109/BigData.2017.8258482.

[37] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper 151, no. 2014 (2014): 1-32.

[38] S. Zhang, J. Rong, and B. Wang, "A privacy protection scheme of smart meter for decentralized smart home environment based on consortium blockchain," International Journal of Electrical Power Energy Systems, vol. 121, p. 106140, Oct. 2020, doi: 10.1016/j.ijepes.2020.106140.

[39] P. Wuille, "BIP 0032." https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki (accessed Jul. 07, 2020).

[40] "BIP 44." https://github.com/bitcoin/bips/blob/master/bip-0044.mediawiki (accessed Jul. 07, 2020).

[41] Hill, B., Chopra, S., Valencourt, P. and Prusty, N., 2018. Blockchain Developer's Guide: Develop smart applications with Blockchain technologies-Ethereum, JavaScript, Hyperledger Fabric, and Corda. Packt Publishing Ltd.

[42] K. Aoki, J. Guo, K. Matusiewicz, Y. Sasaki, and L. Wang, "Preimages for Step-Reduced SHA-2," in Advances in Cryptology – ASIACRYPT 2009, vol. 5912, M. Matsui, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 578–597.

[43] C. Dobraunig, M. Eichlseder, and F. Mendel, "Analysis of SHA-512/224 and SHA-512/256," in Advances in Cryptology – ASIACRYPT 2015, vol. 9453, T. Iwata and J. H. Cheon, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 612–630.

[44] M. Palatinus, P. Rusnak, A. Voisine, and S. Bowe, "BIP 39," Sep. 10, 2013. https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki (accessed Jul. 07, 2020).

[45] Stackexchange, "How long does it take to crack PBKDF2?" https://crypto.stackexchange.com/questions/18173/how-long-does-it-take-to-crack-pbkdf2 (accessed Jul. 07, 2020).

[46] M. Cantu, Joon Kim, and X. Zhang, "Finding hash collisions using MPI on HPC clusters," in 2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, May 2017, pp. 1–6, doi: 10.1109/LISAT.2017.8001961.

[47] F. Buccafurri, V. De Angelis, G. Lax, L. Musarella and A. Russo, "An attribute-based privacy-preserving ethereum solution for service delivery with accountability requirements" in Proceedings of the 14th International Conference on Availability, Reliability and Security, 2019 August, pp. 1-6.

[48] D. Augot, H. Chabanne, O. Clémot and W. George, "Transforming face-to-face identity proofing into anonymous digital identity using the Bitcoin blockchain" in 2017 15th Annual Conference on Privacy, Security and Trust (PST), 2017 Aug 28, pp. 25-2509. IEEE.

[49] W. Shao, J. Chunfu, X. Yunkai, Q. Kefan, G. Yan and H. Yituo, "Attrichain: Decentralized traceable anonymous identities in privacy-preserving permissioned blockchain." in Computers Security 99, 2020, doi: 10.2069.

[50] K. Gurkan, WJ. Koh, and B. Whitehat, "Community Proposal: Semaphore: Zero-Knowledge Signaling on Ethereum", 2020. https://docs.zkproof.org/pages/standards/accepted-workshop3/proposal-semaphore.pdf (accessed Jul. 01, 2021).

[51] M. Mota, "Ethereum HD Wallet", 2021. https://pkg.go.dev/github.com/miguelmota/go-ethereum-hdwallet (accessed Jul. 05, 2021).

[52] A.M. Antonopoulos and G. Wood, "Mastering ethereum: building smart contracts and dapps", O'reilly Media, 2018.

[53] H. Rezaeighaleh, and C.C. Zou, "Deterministic Sub-Wallet for Cryptocurrencies" in 2019 IEEE International Conference on Blockchain (Blockchain), July 2019, pp. 419-424, IEEE.

[54] R. Cox, A. Clements, "Proposal: Go Benchmark Data Format", 2016. https://go.googlesource.com/proposal/+/master/design/14313-benchmark-format.md (accessed Jul. 05, 2021).

XABIER LARRUCEA Dr. Xabier Larrucea received the computer engineering degree in 2002 from the University of the Basque Country, the M.S. degree in software architecture from Deusto University, Spain, in 2003, and a PhD from the University of the Basque Country, Spain, in 2007. He also holds an MBA, and he is PMP certified. He is IEEE Senior Member.

From 2000 to 2002, he was a Research Assistant at IRISA/INRIA in Rennes, France. From 2002 to 2010, he was researcher and project leader covering a wide set of areas related to software engineering. Since 2011 he is project leader at Tecnalia where he led research projects in the area of cybersecurity. His research interest includes software engineering and cybersecurity aspects. He is an Associate Editor of IEEE Access, the IET Software journal, and IEEE Software magazine.

IVÁN GUTIÉRREZ MSc. Iván Gutiérrez received his degree in Computer Science Engineering at UPV/EHU. Expert in Cryptography and Cybersecurity, he owns a master's degree in Software Engineering and Intelligent Systems.

Iván actively contributes to the technology dissemination as ESIC Business Marketing School Professor, as part of the Hyperledger Speakers Bureau and through specialized congresses and learning programs. He is also a peer reviewer for IEEE and SNCS.

After working on neural network sets, ontological driven expert systems and automated image processing, he has grown as a software engineer with technical skills at cybersecurity and software design. As a member of Tecnalia, Iván has been working on RD projects related to PKI, IdM, IAM, chip-based cryptography, contactless technologies, mobile devices and Blockchain.

SERGIO ANGUITA MSc. Sergio Anguita Lorenzo is a MSc Computer Engineering from University of Deusto. From the very beginning, he has worked at DeustoTech Computing on s3lab area, focused on cybersecurity and privacy. He participated in multiple projects for different companies, related to web infrastructures, secure implementation of NFC solutions and massive data analysing for anomalies detection. Sergio also researched in mobile cybersecurity field, automating the process of reversing Android applications looking for malware behavior, threats, or potential risks for user's privacy. He has high knowledge in current technologies and works very well with embedded systems and Linux, being especially focused on this last one and dedicating himself to the areas of analysis, security, and development. Sergio also leads an open source cyber security related project in which he previously researched, and now, he is part of the Tecnalia Cyber Security Research Team, working on the research of Blockchain solutions for industrial systems and processes.

AITOR GÓMEZ-GOIRI PhD. Aitor was awarded a joint BSc in Computing Engineering and Industrial Organization, a MSc in Software Development and a PhD in Computer Engineering at the University of Deusto (Spain).

He started his professional career researching how web and semantic technologies could benefit distributed systems in the Internet of Things. Later, he applied his web expertise to Technology Enhanced Learning projects in the Open University (UK) and afterwards, he worked on enhancing the data collection and analysis of industrial machines with Big Data architectures.

More recently, Aitor has worked in Tecnalia (Spain) using Blockchain solutions to secure and ensure the decentralization, immutability and transparency of data as part of different research projects.

BORJA URQUIZU MSc. Borja Urkizu received his degree in Computer Science Engineering at Deusto University. He owns a MSc in information security. After working on financial systems as a devOps engineering evolving legacy systems he become a software engineering focused on security related products acquiring technical skills on cryptography, software design and performance. Borja also has been involved on products on the gaming domain as a senior backend engineering working with real-time technologies and security challenges on gaming platforms. As a member of Tecnalia, Borja has been working on RD projects related to PKI, IdM, IAM, Hyperledger Fabric, Ethereum and Decentralized Privacy-Preserving Proximity technology.