Discovering missing me edges across social networks

https://doi.org/10.1016/j.ins.2021.01.004

A Blockchain-based Approach for Matching Desired and Real Privacy Settings of Social Network Users

Gianluca Lax, Antonia Russo, Lara Saidia Fascì

University of Reggio Calabria, Italy
lax@unirc.it, antonia.russo@unirc.it, fascilara@gmail.com

Abstract

Social networks store a considerable amount of personal data, which are also a source of information for business. To comply with users' privacy rights, all social networks allow users to select the level of privacy they desire. However, what occurs if the privacy choices of a user are modified unilaterally by the social network? The privacy settings chosen by the user are stored by the social network, which acts as a *privileged party*, which could tamper with the user's choices at any time. This paper addresses this problem and proposes a decentralized approach to manage the privacy settings of a user. Any change in the privacy settings of a social network user is validated by a smart contract to ensure that it is compliant with users' expectations. The proposed solution has been implemented as an Ethereum-based decentralized application to validate the effectiveness of the proposed approach.

Keywords:
Facebook, Instagram, smart contract, Ethereum, DApp, accountability, GDPR

1. Introduction

The amount of personal data available on the Web has grown exponentially, and a concentration of such data is placed in the most prominent social networks, such as Facebook, Instagram, Twitter, YouTube, and LinkedIn. The availability of a massive amount of personal information has raised privacy concerns about how social networks manage these data [14, 38, 47]. Conversely, data are an essential asset for social networks: they can sell personalized advertising with characteristics that match those of users because

https://doi.org/10.1016/j.ins.2021.01.004

they know users' demographic information, activities, purchasing patterns, interests, and much more information. The contrast between the need to preserve users' data privacy and the need for social networks to monetize users' data is evident: for this reason, social networks allow users to choose to what extent they want their data to be disclosed.

However, allowing a user to choose the desired privacy settings may not be sufficient; for example, in 2018, the discovery that Facebook gave access to the personal data of more than 87 million users to Cambridge Analytica fueled interest in the risks of privacy violations [28]. In 2019, a study performed by the European Commission [44] concluded that many people do not know how to change privacy settings in social networks.

Concerning the problem of privacy settings in social networks, several studies have been performed, primarily focusing on enabling users to specify correct privacy requirements [41, 12, 40, 39] and detecting incorrect sharing settings [34, 7, 32]. In this paper, we manage a new issue related to the user's privacy settings in social networks. Although a user can set whether personal data can be transferred to third parties and which data can be displayed to whom via the Internet, the user's choices are saved by the social network and used to manage his/her data accordingly. However, what occurs if the choices of the user are modified unilaterally by the social network? Consider the following hypothetical example. *John* accuses the social network *Fantasy* of having permitted a third party to access his data and asks for compensation. *Fantasy* replies by showing the permission given by *John* to share his data. At this point, such a dispute cannot be resolved: *Fantasy* could have tampered with the stored privacy settings of *John* to hide a data breach; on the other hand, *John* could have simulated this privacy violation to obtain compensation. This issue occurs because the privacy settings chosen by a user are stored by the social network, which acts as a *privileged party*, which could modify the user's choices at any time. To the best of our knowledge, no solution to this problem has been proposed in the literature. Existing approaches based on data encryption or decentralized storage [13, 25, 9] cannot be used for the most popular social networks. Indeed, social networks show their service for free and generate most of their revenue by offering custom advertisements to specific types of consumers [8, 43]. Because the advertisement-based business model of social networks strongly depends on user data, any approach aimed at hiding such data from social networks, such as schemes based on data encryption or decentralized storage, is not effective in practice.

https://doi.org/10.1016/j.ins.2021.01.004

Our paper proposes a solution to this problem that is based on the decentralized storage of users' privacy settings in such a way that the social network cannot modify such settings without this change being tracked. The proposed approach is based on *blockchain*, a recent disruptive technology that has already been applied in many fields, such as finance and smart cities. The blockchain is a fully distributed cryptographic system that guarantees transparency, traceability, and immutability of registered information. The control is distributed among several nodes in a peer-to-peer fashion and ensures that transactions comply with programmable rules in the form of *smart contracts* [31]. The blockchain is in charge of storing both the privacy preferences of a user and the privacy settings assigned to him/her by a social network. Moreover, using suitable smart contract functions, the blockchain determines whether the privacy settings assigned to the user by the social network are compliant with those declared by the user.

A further benefit derived from the use of the proposed approach is that it favors the achievement of one of the goals of the recent General Data Protection Regulation (GDPR) [19], the new European Union privacy law. The GDPR is intended to apply guidelines and regulations to how data are analyzed, managed, stored or exchanged and applies to organizations that are registered in the EU, organizations that have an establishment or subsidiary in the EU, and to any organization that sells goods or shows services and must process or track the personal data of EU residents. The GDPR places strict obligations in terms of accountability on organizations to show their compliance with the regulation and also implies that organizations will have to maintain written records of the processing activities they perform. The use of the proposed solution achieves accountability because it allows a social network to show the correct management of the user's privacy choices. Indeed, such choices are not self-certified by the social network, as currently occurs; however, they are stored in a decentralized way on the blockchain that guarantees the integrity and authenticity of data.

The proposed approach does not aim to prevent a social network from violating the user's privacy but only to detect whether a violation has occurred in case of dispute. For example, in the case of *John* and *Fantasy* introduced above, the use of the proposed scheme allows anyone to determine whether the social network misbehaved based on information publicly available on the blockchain. The proposed approach is designed in such a way that the information stored in the blockchain does not show any sensitive information about the user.

https://doi.org/10.1016/j.ins.2021.01.004

The primary contributions of this paper are:

1. we identify an issue in the management of the user's privacy settings that allows a social network to perform and hide a privacy breach;
2. we propose a blockchain-based scheme for detecting the malicious behavior of a social network that exploits this issue;
3. we analyze the privacy settings of five prominent social networks (Facebook, Instagram, Twitter, YouTube, and LinkedIn) and define a graph-based model to represent the concepts and relations of privacy settings concisely;
4. we implement the proposed approach for Facebook, Instagram, Twitter, YouTube, and LinkedIn to demonstrate its effectiveness.

This paper is structured as follows. In the next section, we discuss how the most commonly used social networks manage the privacy preferences of users. In Section 3, we introduce a model for defining the privacy settings of a user in a social network and show how the model is instantiated in the most popular social networks. This model is used to define both the privacy settings desired by a user and those stored by the social network. In Section 4, we present the approach proposed to solve the privacy issue faced in this study. In Section 5, we instantiate the proposed approach in an example scenario, and we show the technical details about how our solution has been implemented. Related work is discussed in Section 6. Section 7 presents the advantages and limitations of the proposed solution. Finally, in Section 8, we draw conclusions and propose future work.

2. Privacy Settings in Social Networks

In this section, we describe the privacy choices that a user can make in the most used social networks.

We start with the user's privacy settings on Facebook [17]. The "Privacy Setting and Tools" section allows users to choose who can see their activity (future posts). A user can choose among several levels of visibility of posts: public, available to friends on Facebook, available to friends on Facebook except for certain selected users, available to a specific group of friends, available to a custom group of people, or private. Also, users can choose who can find their profile. To be more specific, a user can choose if anyone can send a friend request or if the sender must be friends with at least one of the user's friends. Other privacy settings in this section include the

https://doi.org/10.1016/j.ins.2021.01.004

possibility of choosing whether other users can see account information such as the user's friends list, phone number, and email address. The friends' list can be public, private, or available to friends of friends; the user's phone number and email address can be set to be available to everyone, friends, or friends of friends. In the "Timeline and Tagging Settings" section, users are shown more privacy settings that manage their profile privacy. Users can filter comments and decide whether to allow their friends to post on their timeline. They can also choose who can see what others post on their timeline and whether to allow other users to share their posts to stories or not. Also, this section displays tagging options: users can choose who can see posts they are tagged in, and they can also decide if they want to review tags before they are submitted. Facebook also allows its users to block specific users, apps, or pages from interacting with their profile or from performing specific actions. In addition to sharing content, Facebook users can also chat: the privacy options associated with this feature allow users to choose if they want to show their activity status or not. Facebook users can even change their story privacy settings, and Facebook stories can be public, available to friends and connections, available to friends, or available to a custom set of users. Last, location services, sometimes called location access, are available in Facebooks mobile app and help Facebook show its users location-based features, including allowing them to post content that is tagged with their location, obtain more relevant ads, find places and Wi-Fi nearby, and find nearby friends. When location services is on, a user can choose to turn Background Location on or off, which allows Facebook to access the device's precise location when the user is not using the app or not.

On Instagram [27], users can manage their privacy settings on different levels. First, users can manage account privacy, where anyone can view a user's profile and posts on Instagram by default. Users can decide to make their profile private so that only approved followers will be able to see its content. If a user's posts are set to private, only approved followers will see them on hashtag or location pages. Second, users can decide how photos and videos depicting them are added to their profile by the "Photos of you" section: they can choose to add them automatically to their "photos and videos of you" or not. If a user's profile is public, people can reshare the user's posts on their stories. There is an option to turn this feature off through the "Resharing stories" section. Similar to Facebook, Instagram allows users to block specific accounts from viewing their profile. Blocked accounts cannot view any sort of content posted by the user who blocked them. Instagram

https://doi.org/10.1016/j.ins.2021.01.004

users can also decide if they want their activity status to be shown or not, and they can decide who can see their stories. Users can choose with whom to share their stories. Everyone on Instagram can send direct messages to any user, whether they follow them or not. However, messages from people other than one's followers are kept under a different section (Requests) in direct messages. While Instagram does not let users stop direct messages for regular messages, it can restrict direct message replies for stories. Instagram shows three privacy options for message replies to stories: Everyone, People you follow, and Off. Last, in the "comment controls" section, users can filter comments and choose who can post comments on their posts. Depending on what the user chooses, comments can be posted by Everyone, followed accounts, or followers.

In Twitter [46], privacy settings are categorized into three sections. The first section is about Tweets, which can be set to be public or protected. Public tweets, which is the default setting, are visible to anyone, including people who do not have a Twitter account, and protected tweets are visible only to followers. The second option in this section addresses the location of Tweets, which enables the addition of precise location information to a Tweet. This feature is off by default. When it is enabled, it allows Twitter to collect, store, and use Tweets' precise locations obtained by GPS. The last option in this section addresses allowing people to tag users in photos: this can be set to anyone, only following, or nobody. The second section is Direct Messages: by default, users can receive a private conversation request only by who follows them. However, a user can choose to receive requests from anyone on Twitter. The second option of this section allows users to turn on or off the notification that they have seen a message: by turning off this setting, a user would not be able to see read receipts from others. The third section, called Discoverability, allows others to find the user by email address or phone number.

The privacy settings on YouTube are simple: these settings allow users to choose who can see their liked videos, saved playlists, and subscriptions. By default, such resources are public, but a user can maintain one or more categories of resources private.

LinkedIn shows privacy settings in three sections: profile and network information, LinkedIn activity, and job-seeking preferences. Users can choose their profile visibility for viewers not logged in LinkedIn, and the profile's public visibility can be turned on or off. From this level, privacy settings on personal information, posts, or activities are generated; by default, LinkedIn

https://doi.org/10.1016/j.ins.2021.01.004

sets primary information as public, but users can decide what category of data include in the public profile. Public profiles can be found through search engines. Public visibility can be switched on or off: if it is off, the profile will not be visible for not logged-in members; if on, users will show basic profile information (name, number of connections, industry, and region). Profile photos can be shared with connections, the network (LinkedIn members connected up to three degrees of separation), and anyone (all LinkedIn members). Users can choose to show their headline or not, posts and activities, current experience, past experiences, and education in the profile. Also, users can choose to show only the first letter of the last name. Personal information, such as email, can be shared with no one, 1st-degree connections, 1st- and 2nd-degree connections, and anyone on LinkedIn. With regard to connections, users can choose to hide or not hide them from other connections. LinkedIn also implements the concept of "views": if a user sees another user's profile, the user with the viewed profile will be notified. LinkedIn protects the user's privacy via three different levels of profile viewing options: name and headline, private profile characteristics, and private mode. Concurrently, a user can manage active status by three options: no one, connections, and all LinkedIn members. Users can choose whether or not they want to share changes in jobs or education with the network and to notify the network if they have been mentioned in a blog or article post. Users can allow others to be mentioned or tagged in content, such as posts, comments, and tags in the photo. Users can decide whether LinkedIn can save the information entered when applying to jobs (internal or external application) directly on LinkedIn. When users apply for a job, they can choose to share their full profile with the job poster. Users can be open to opportunities or not: in the first case, recruiters will be able to find users by career interests. Users can also decide to create a job alert for companies: they will be notified of new jobs matching their skills. Last, users can choose to share or not share their interests with recruiters.

The description of the choices that social networks show about privacy is used in the next section to define a model to represent the privacy settings of a user.

3. Modeling Privacy Settings

The purpose of this section is to define a model to represent the privacy settings of a user in a social network. To do this, we introduce certain

https://doi.org/10.1016/j.ins.2021.01.004

preliminary definitions.

Definition 1. *Given a social network S, the* privacy feature *graph of S is a direct graph $PF^S = \langle N, E \rangle$, in which nodes are divided into two disjoint and independent sets G (groups) and R (resources); thus, $N = G \cup R$. A node in G represents a group of users (i.e., profiles) of the social network S, while a node in R represents a type of information in a user's profile. Given two nodes g_i, $g_j \in G$ with $i \neq j$, an edge from g_i to g_j denotes that all the users in g_j are also in g_i.*

Now, we show an example to help the reader understand this definition better.

Example 1. *In Figure 1, the privacy feature graph of Instagram is shown. In this graph, we have four group nodes $g_1, \ldots, g_4$ (i.e., $|G| = 4$) and five resource nodes $r_1 \ldots r_5$ (i.e., $|R| = 5$). As described in Section 2, an Instagram user can choose to show profile data to four categories of users: g_1 denotes all Instagram users; g_2 and g_3 denote the follower users and followers who are also followed by the account owner respectively; and g_4 denotes all the non-follower users who are followed by the account owner. Again, we have seen that on Instagram, privacy settings apply to five categories of resources: $r_1, \ldots, r_5$ denote photos and videos, stories, story message replies, comments, and active status, respectively. Finally, the edges from g_1 to g_2, g_3, and g_4 denote that g_1 is a superset of all the other group nodes (e.g., followers are included in everyone). Again, for the same reason, note that g_2 is a superset of g_3.*

Starting from the analysis shown in Section 2, we show how it is possible to build a privacy feature graph for each of the social networks considered in the previous section.

Facebook has five group nodes: everyone, friends, friends of friends, restricted users, and custom (the user can decide to include or exclude certain users). Resource nodes are made of posts, tagged posts, stories, friends' lists, profile information, and app information.

In Twitter, we identify three group nodes (everyone, everyone in Twitter, and followers) and the following resource nodes: tweets, location, tag, direct messages, notification, email, and phone number.

YouTube is simplest and only has the group node everyone and three resources: videos, saved playlists, and subscriptions.

https://doi.org/10.1016/j.ins.2021.01.004

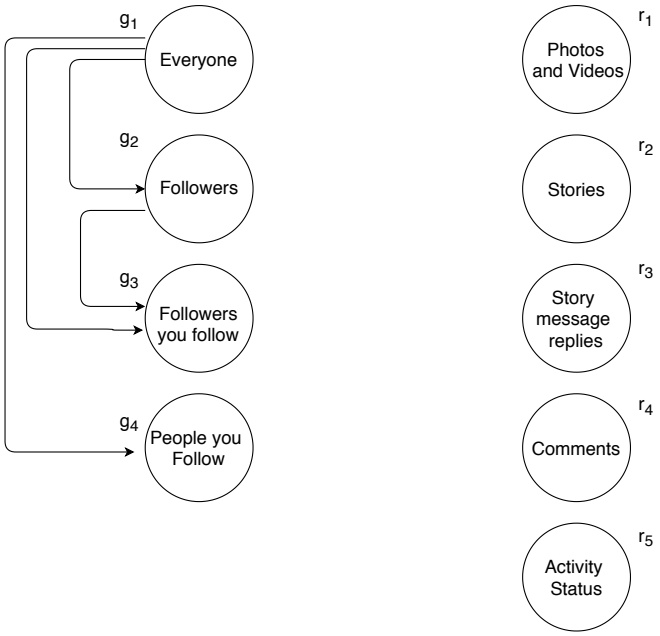

Figure 1: Privacy feature graph of Instagram.

In LinkedIn, there are three group nodes (everyone, connections, and members), and there are five resource nodes (posts, tagged posts, personal information, job applications, and job interests).

The model defined in this study can be extended to many other social networks. Now, we define a model for the privacy settings that a user can choose in a social network.

Definition 2. *Given a user U and a social network S with privacy feature graph $PF^S = \langle N, E \rangle$, the* privacy setting *graph of U in S is a direct graph $PS^S_U = \langle N, E \cup P \rangle$, where $p \in P$ is an edge $(g \in G, r \in R)$.*

This graph has the same nodes as PF^S, and a superset of edges: the additional edges w.r.t. PF^S are edges from a group node to a resource node. Specifically, an edge $(g \in G, r \in R)$ models that the users denoted by g can access the information shown by the resource r.

Example 2. *Considering Instagram again, we examine the privacy settings of three different users mapped by three privacy setting graphs $PS^S_{U_1}$, $PS^S_{U_2}$,*

https://doi.org/10.1016/j.ins.2021.01.004

and $PS^S_{U_3}$.

The first user U_1 *has a public profile. Then,* $PS^S_{U_1}$ *is similar to* PF^S *and has four edges* $(g_1, r_1), \ldots, (g_1, r_4)$*, which indicates that everyone* (g_1) *can access all resources* $r_1 \ldots r_4$.

In the second case, user U_2 *chooses to set her/his profile as private. In a private profile, only* U_2*'s followers can access resources; therefore,* $PS^S_{U_2}$ *has as edges* $(g_2, r_1), \ldots, (g_2, r_4)$*, where* g_2 *are the followers of* U_2.

The most interesting and probably common case is when a user, say U_3*, has customized privacy settings modeled by the privacy setting graph* $PS^S_{U_3}$*, such as the one shown in Figure 2. The edge* (g_1, r_1) *represents that every Instagram user can access photos and videos posted by* U_3*;* (g_1, r_2) *represents that every Instagram user can access stories posted by* U_3*. Story message replies can be sent by those users who are followed by* U_3*; this is shown through the edges* (g_3, r_3) *and* (g_4, r_3)*. Permission to comment is given to followers and followed accounts (i.e., "People you follow and your followers"); this is shown by the edges* (g_2, r_4)*. and* (g_4, r_4)*. Finally,* U_3 *has chosen not to show the activity status: this is modeled by the lack of edges to the node activity status.*

Now, we are ready to introduce the two definitions that will be widely used in the following.

Definition 3. *Given a privacy setting graph* $PS^S_U = \langle G \cup R, E \cup P \rangle$*, we define the* extended privacy setting *graph* $EPS^S_U = \langle G \cup R, P \cup P' \rangle$ *as the bipartite graph such that every edge connects a vertex in* G *to one in* R*, and* P' *is the set of edges* (g_i, r_j) *such that there exist in* PS^S_U *the edges* (g_y, g_i) *and* (g_y, r_j) *with* $i \neq y$.

The following example shows an intuitive idea of how to build an extended privacy setting graph.

Example 3. *Consider the privacy setting graph shown in Figure 2. To obtain the bipartite graph, we must remove the edges between group nodes because they violate the bipartite requirement. For any edge* $e \in E$ *from* g_y *to* g_j*, if* g_y *has no edge to any resource node, then* e *can be removed. Otherwise, for any edge* $(g_y, r_j) \in P$*, we add to* P' *a new edge* (g_j, r_j) *before removing* e*. Figure 3 shows the extended privacy setting graph derived from the privacy setting graph shown in Figure 2. For example, note that everyone can access* r_1 *and* r_2*, and everyone is a superset of* g_2*, leading to the addition of the edges* (g_2, r_1) *and* (g_2, r_2)*, when the edge* (g_1, g_2) *is removed.*

https://doi.org/10.1016/j.ins.2021.01.004

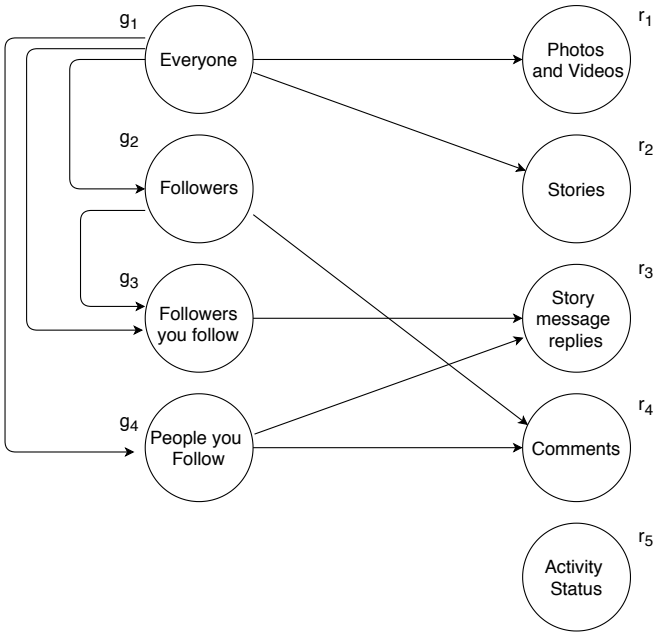

Figure 2: Privacy setting graph of a customized profile.

We conclude this section by defining a binary encoding of the model, which will be useful in the implementation of the proposal.

Definition 4. *Given an extended privacy setting graph* $EPS_U^S = \langle G \cup R, E \cup P \rangle$*, we define the* serialized privacy setting SPS_U^S *as the* $|G| \cdot |R|$*-bit string such that the* x*-th bit with* $1 \leq x \leq |G| \cdot |R|$ *is* 1 *if and only if there exists the edge* (n_i, r_j) *with* $i = (x-1)/|G| + 1$ *and* $j = (x-1)\%|G| + 1$*, where* / *and* % *denote the quotient and the remainder of Euclidean division, respectively.*

This definition allows us to represent an extended privacy setting graph using a bit string. In the following example, we show how this string is obtained.

Example 4. *Consider the extended privacy setting graph shown in Figure 3. This graph can be shown by the adjacency matrix reported in Table 1, whose 32-bit string encoding is* $0x000FF370$*, based on the definition given above.*

https://doi.org/10.1016/j.ins.2021.01.004

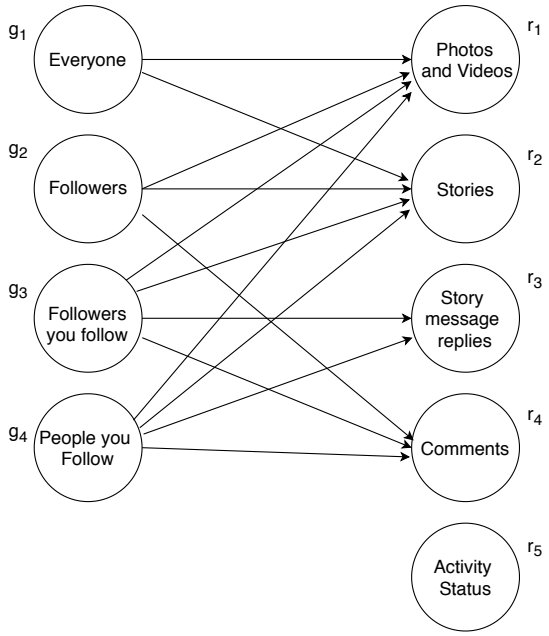

Figure 3: Extended privacy setting graph.

4. Proposed Solution

In this section, we present the proposed approach to ensure that the privacy settings requested by a user cannot be modified by a social network without being detected.

We introduce a scenario in which we have four actors:

- an *online social network*, a decentralized and distributed computer network that shows services through the Internet.
- a *user*, a person using one social network to communicate with other people and share information and resources.
- a *Blockchain*, a Distributed Ledger enabling smart contracts, such as Ethereum.
- a *smart contract*, which is deployed on the blockchain.

https://doi.org/10.1016/j.ins.2021.01.004

	g_1	g_2	g_3	g_4
r_1	1	1	1	1
r_2	1	1	1	1
r_3	0	0	1	1
r_4	0	1	1	1
r_5	0	0	0	0

Table 1: Adjacent matrix of the considered SPS_U^S.

Figure 4 shows the architecture of the proposed solution and the interactions among the actors performed based on the following operations.

1. *Social network registration.* Each social network needs an *External Owned Account* (EOA) to operate on the blockchain, and a blockchain address is generated as follows: a pair of private and public blockchain keys are generated, and then, the corresponding blockchain address is computed as the cryptographic hash of the public key. Each social network makes its blockchain address publicly available.
2. *User registration.* A user also needs an EOA: following the same procedure described above, each user U can generate her/his blockchain address, say A. Also, each social network SN includes SN includes two new fields in the user's profile: the first is for the user's blockchain address, the second is filled in by the user's secret (e.g., a password), which works as a *salt*. Finally, U generates a transaction on the blockchain from A to the social network blockchain address, which is publicly available, with a payload $H(snid, A, salt)$, where *snid* is the ID of the user in the social network, and H is a suitable cryptographic hash function. Via this transaction, the user links her/his social network identifier to blockchain address A. This mapping can be verified only by knowing *salt*, and this is performed by the social network.
3. *User verification.* This operation is performed by a social network to verify that the blockchain address declared by one of its users is correct. Given a user U, this check is performed as follows: first, the social network extracts from U's profile the values *snid*, A, and *salt*. Then, it searches for a received transaction coming from address A having in

https://doi.org/10.1016/j.ins.2021.01.004

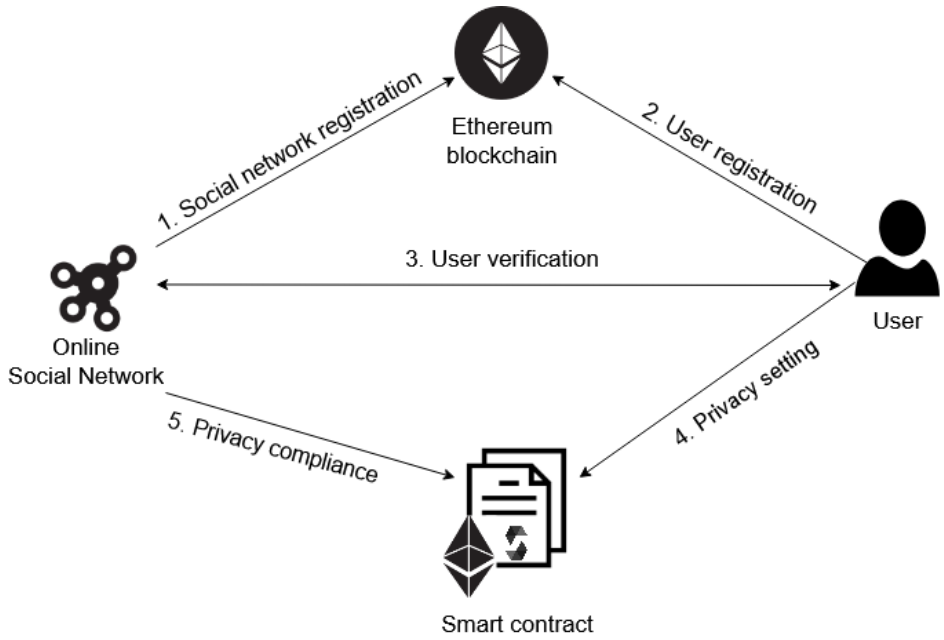

Figure 4: Interactions between the actors.

payload $H(snid, A, salt)$. If no transaction is found, then this check fails; otherwise, the mapping is correct.

4. *Privacy setting.* This operation is performed by a user to declare her/his desired privacy settings. Specifically, this is performed by calling the function `setPrivacy` of the smart contract (say SM) and by passing the extended privacy setting bit string as a parameter (see Definition 4) derived by the privacy feature graph that represents the desired privacy settings of the user (see Definition 2). The smart contract stores the settings associated with the user's blockchain address.
5. *Privacy compliance.* When a social network wants to assign or modify the privacy settings of user U, the function `checkSettingGraph` of SM is called to ensure that the new settings are compliant with the user's preferences stored on the blockchain,
   This function has A and B_U, where A is the blockchain address of the user, and B_U is the extended privacy setting bit string derived from the privacy feature graph representing the privacy settings to be assigned to the user. This function (1) extracts the bit string P representing the preferences of the user saved locally, if any, and (2) compares bitwise

https://doi.org/10.1016/j.ins.2021.01.004

P and B_U to verify that a zero bit in P is associated with a zero bit in B_U. Only if this check does not fail, the function returns that the settings are compliant with the user's preferences.

In summary, the idea underlying the proposed approach is to exploit a blockchain smart contract to store both the desired privacy preferences of the user and the privacy settings assigned to the user by the social network. Also, the smart contract verifies whether the privacy settings assigned to the user by the social network are compliant with those declared by the user. Because all operations are stored in the blockchain, and transactions cannot be modified, this solution implements an easy method to show accountability of all the privacy assignments performed by a social network. A social network can use this solution as proof of having acted correctly, based on the accountability requirement of the recent General Data Protection Regulation (GDPR) [19].

5. Design and Implementation

In this section, we describe the development of the proposed solution, which is based on a decentralized application (DApp), called *Your Privacy Manager DApp*, which runs on a blockchain. We start by discussing the choices concerning the architecture of the system.

5.1. System architecture

We start by surveying the most commonly used blockchain technologies and highlighting their advantages and drawbacks based on the interesting analysis reported in [42].

Ethereum [49] is considered the second largest and global cryptocurrency platform and is a permissionless blockchain enabling the creation of decentralized applications through the use of smart contracts. This platform can be considered a system that is globally shared and implementing a cryptographically secure transaction-based state machine [49]. A smart contract is defined as a piece of code verifying and enforcing conditions that stipulate a digital contract between parties that does not require a third intermediary. Smart contracts are written in *Solidity*, an object-oriented and high-level Turing complete programming language.

IOTA [36] networks were designed for IoT applications and are permissionless blockchain networks that are built on Tangle, a new data structure

https://doi.org/10.1016/j.ins.2021.01.004

based on a directed acyclic graph, which does not need blocks, miners, or any chain. For this reason, IOTA transactions are free. Although this platform can yield better performance than Ethereum and Hyperledger blockchains, its primary limitation is that devices may not be not capable of performing the Proof of Work, resulting in a bottleneck when transactions occur [18].

EOSIO [16] is the first blockchain platform that uses the Delegated Proof of Stake consensus algorithm. Converse to traditional proof-of-work-based systems, EOSIO is public, permissionless, and suffers from serious attacks derived from exploiting vulnerabilities in DApps and leading to millions of dollars lost for EOSIO users, as discussed in references [26, 37].

Among permissioned blockchains, Hyperledger Fabric [2] is an implementation of an open-source private blockchain running smart contracts and is intended to form a foundation for developing applications with a modular architecture. Hyperledger Fabric allows components, such as consensus and membership services, to be plug-and-play.

MultiChain [20] is a platform enabling the creation and deployment of private blockchains to be developed and used inside organizations. The system administrator sets a series of user permissions to introduce controls over transactions and block size.

Chain Core [15] is another platform for private blockchains and is typically used to initiate and transfer financial assets based on permission from the blockchain infrastructure. Corda [10] is a distributed ledger platform for recording and enforcing business agreements among institutions. Chain Core and Corda also rely on smart contracts and allow participants to manage permissions.

Open Chain [35] is an open-source distributed ledger technology based on the Partitioned Consensus: every Open Chain instance has one authority validating transactions. This platform aims to manage the digital assets of organizations in a scalable and secure way.

This analysis provides a way to verify the suitability of each technology with respect to the needs of the proposed solution. The Hyperledger Fabric, MultiChain, Chain Core, and Open Chain platforms suffer from limitations due to their permissioned nature and goals [2, 15, 10, 35]: users belong to disparate environments and organizations, which makes it difficult to individuate who can arrange permissions [20].

Therefore, from the perspective of users' accessibility, Ethereum, IOTA, and EOSIO are valid options for the implementation of the proposed solution. Among these platforms, we choose Ethereum for developing the proposed

https://doi.org/10.1016/j.ins.2021.01.004

solution because it is the most used and widespread permissionless Blockchain enabling smart contracts.

5.2. Implementation

In this section, we describe the implementation of the DApp called *Your Privacy Manager DApp*. This DApp provides a front-end to interact with the Ethereum blockchain via a smart contract, and the application's back-end code is executed on a peer-to-peer decentralized network.

The development environment relies on Truffle, a testing framework for blockchain that uses the Ethereum Virtual Machine (EVM)), which simplifies the DApp implementation process for developers.

For the deployment of the smart contract, Ganache is used. Ganache is part of the Truffle suite and is a personal blockchain used to develop distributed applications for Ethereum and Corda. Additionally, Ganache can be used to run tests and deploy contracts. The local test network produced by Ganache can be used for development purposes only. To deploy the smart contract on the Ethereum blockchain, a connection to the primary network is necessary.

A user can interact with the Ethereum decentralized application through the browser without the need to run a full Ethereum node but does need MetaMask. Unlike other wallets, MetaMask is a web browser plug-in that supports Brave, Google Chrome, and Firefox and provides a user interface to sign blockchain transactions and to manage identities.

For the implementation of the proposed application, we used Node.js, an open-source, cross-platform JavaScript run-time environment. The use of Node.js enables the use of the same programming language to develop both server- and client-side scripts.

The user interface consists of HTML and JQuery functions that display different DOM elements depending on what options the user selects. The core of the client-side application is a JavaScript file that contains all the functions necessary to load the smart contract, connect with the wallet, capture the user's desired privacy settings and call the smart contract functions.

Figure 5 schematizes the sequence diagram of the privacy setup process. A similar diagram can be traced to represent the verification of compliance. A user accesses the DApp using a web browser with Metamask and submits the desired privacy settings through a form. Then, the JavaScript asynchronous function `setPrivacy()` calls the setPrivacy function of the smart contract.

https://doi.org/10.1016/j.ins.2021.01.004

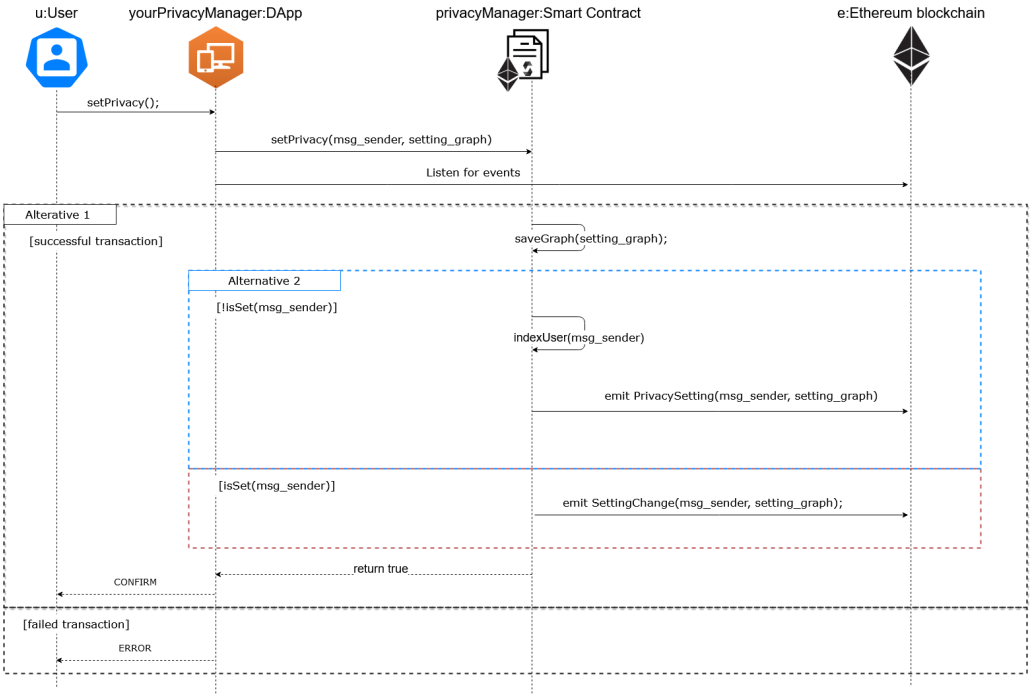

Figure 5: Sequence diagram showing functional calls and events.

The smart contract then saves the setting graph of the user (sender): Alternative 2 in Figure 5 records if this is the first time the user sets his/her privacy, and how smart contract associates this graph with the new user. The result of this operation is logged on the blockchain using the proper event. Alternative 1 represents the result of the operation requested by the user: if no error occurred in the process, a confirmation message is sent to the user. An error message is shown to the user in the case the transactions failed for any reason (e.g., out of gas, incorrect signature, exceeding block gas limit).

Now, we show the implementation of the privacy settings in the case of an Instagram user. We present the data flow among several actors: the user, its Dapp, the social network Instagram, and the Ethereum blockchain with the smart contract.

A preliminary operation that occurs only once regards the link between the user's Instagram account and blockchain address. For this purpose, the DApp includes three fields to be filled in by the user:

- *Username*, which is the username of the user on Instagram;

https://doi.org/10.1016/j.ins.2021.01.004

- *Blockchain address*, which is set to the user's blockchain account address, say A;
- *Password*, which is set to the password of the user on Instagram.

Once the user fills in these fields, the application generates a transaction to the social network blockchain address with a payload $\langle H(username, A, password^1)\rangle$.

The social network receives this transaction, extracts the blockchain address of the sender (i.e., A), searches for the username who has declared this address, and retrieves the password of this user. Now, the social network has all the data required to calculate the same digest and to verify that the digest received by the transaction has been correctly generated. This process assures that the link between the username and blockchain address has been sent by the user. To generate such a transaction, the password of the user and the private key associated with the blockchain address must be known because the transaction is signed.

Then, the user can select their desired privacy setting on the DApp. The part of the user interface relative to this operation is shown in Figure 6, which shows privacy settings that the user can choose. After having selected the desired options and input their password by clicking the button Submit, the DApp generates an Ethereum transaction for the smart contract. This transaction is built based on the operation described in Section 3 and is handled by the function called `PrivacyManager`; the code of this function and the entire smart contract is reported in Figure 7. The smart contract is implemented in Solidity, a statically typed programming language designed for developing smart contracts, compiled to bytecode, and executed on a suitable virtual machine (EVM). The smart contract has been written using the CRUD pattern. Considering how the EVM currently works, this process is recommended. In the first portion of the code, a structure, `UserSettingStruct`, is defined using the struct keyword. This structure contains two types of information:

- *setting_graph*, of type `bytes`, is a dynamically sized array used to represent the setting graph.

[1]For the sake of presentation, we assume that the OSN knows the user's password. Actually, only the password digest is known: thus, the password digest should be used in place of the password.

https://doi.org/10.1016/j.ins.2021.01.004

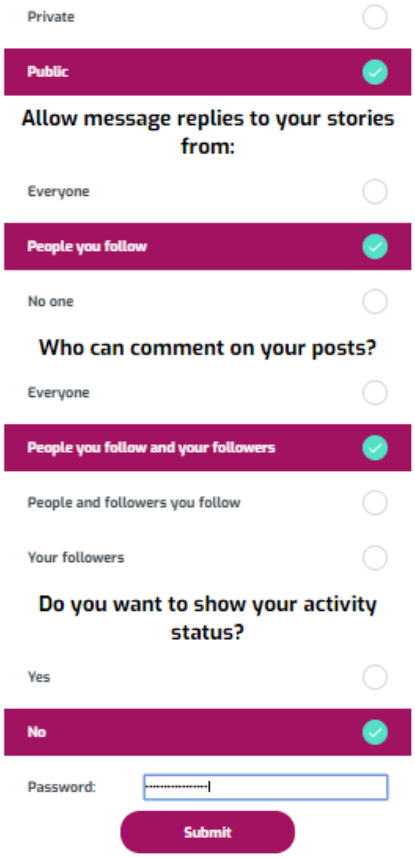

Figure 6: Selection of the desired privacy settings on the DApp.

- *index*, of type `uint`, represents the position of a user's blockchain address in the `userSettingIndex` array, based on the CRUD pattern.

Line 8 defines a mapping between the address of a user and an element of the previously defined structure. Finally, the array `userSettingIndex` is used to verify the correctness of this mapping.

The following events are also part of the smart contract:

- *PrivacySetting*: This event is emitted every time a user declares a desired privacy setting for the first time;
- *SettingChange*: This event is emitted every time a user updates the desired privacy settings;

https://doi.org/10.1016/j.ins.2021.01.004

- *CheckSetting:* This event is emitted whenever a privacy setup is configured through the OSN front-end and shows whether the privacy setting graph received as an input is compliant with the one previously set by the user.

The function `isSet` (Lines 14-17) checks if a user is associated with a setting graph. To do this, the function first checks if the `userSettingIndex` array is empty (Line 15). If the array is empty, no association exists, and the function returns false. If the array is not empty, the function retrieves the index associated with the input from the `userSettingStruct` struct through the mapping. Then, the function retrieves the address contained in the `userSettingIndex` array at the retrieved index position and checks if this address matches the input. A Boolean value is returned based on a comparison result.

The function `setPrivacy` (Lines 18-30) is used by users to set or update their privacy preferences. The input `_setting_graph` is a dynamically sized array of bytes used to represent the privacy setting graph associated with the user's preferences (see Definition 4). First, the `isSet` function is called (Line 19). If `isSet` returns false, the function indicates that the sender has never submitted a graph; thus, the new `setting_graph` value is stored as the value of `setting_graph` in the `userSettingStruct` struct associated with the address of the sender through the mapping (Line 21). Then, the address of the sender is pushed in the array `userSettingIndex`, and the new index is stored as a value for the `index` attribute in the structure. Finally, the `PrivacySetting` event is emitted.

Everything works similarly if the user had already submitted a graph in the past (Lines 25-29). In this case, only an update is required. Thus, only the graph contained in the structure must change, and the index remains the same. Once the update has occurred, the `SettingChange` event is emitted.

The function `checkSettingGraph` (Lines 31-50) is called by an OSN interested in associating a set of privacy options with a user. First, the function checks if the user address has ever been associated with a privacy preference graph. To do this, the `isSet` function is called. If the result of this call is false, the `CheckSetting` event is emitted, and the value of its third parameter is false because the user has never defined her/his privacy preferences. Otherwise, if `isSet` returns true, the `checkSettingGraph` function checks if the input privacy settings and those already associated with the user are compatible in size. If their sizes do not match, then the `CheckSetting` event

https://doi.org/10.1016/j.ins.2021.01.004

is emitted, and the value of its third parameter is set to false. Otherwise, the function compares the input setting graph `request` coming from the OSN with the user's request. This comparison is performed for each corresponding byte of the byte arrays by a bitwise OR: if the output of this OR operation is greater than the byte in the stored `setting_graph` in the `userSettingStructs`, then the settings are not compliant with the user's preferences. Eventually, a `CheckSetting` event that displays the OSN address (`msg.sender`), the user's address, and the result of the call is emitted: in particular, the result will be true if the request from the OSN is compliant with the user's, false otherwise.

Finally, we consider the case in which a user wants to update the privacy setting. This case is similar to the setting of the privacy seen above, with the only difference that instead of the `PrivacySetting` event, the `SettingChange` event is emitted. The use of events is necessary because the real value returned by a function is always the hash of the transaction that is created: transactions do not return a value to the front end because they are not immediately mined and included in the blockchain. As a solution, to obtain the return value from the function, the front end must maintain watching for that event. This process also implies that a listener must be active in detecting the events. Specifically, the OSN listens to the events, and when the `SettingChange` event occurs, the social network checks if the new privacy settings of the users are compliant with the existing settings associated with the user. If they are not compliant, then the social network has to update the user settings accordingly.

To validate the proposed smart contract, we used Etherscan, a Block Explorer and Analytics Platform for Ethereum, where it is possible to find all Ethereum transactions. When a function of the smart contract is called, the transaction begins, and after a few seconds, it is visible on Etherscan. Also, in the *Event Logs* section, the events emitted during the execution of the function are also shown. The contract has been implemented by *Remix - Solidity IDE* connected with Metamask, an extension for accessing Ethereum enabled distributed applications from the browser: the Ethereum testnet used is *Ropsten*.

The implementation of the proposed solution can be found on GitHub at `https://github.com/Lara-F/Your-Privacy-Manager`. The smart contract has been deployed on Ropsten and is reported at `https://ropsten.etherscan.io/address/0x3657b2322f2dde1fea4af963ae2f5b7837db4fe1`.

https://doi.org/10.1016/j.ins.2021.01.004

6. Related work

In this section, we survey the most important proposals related to the proposed approach. We start by considering the literature related to social networks.

Facebook, Twitter, and Instagram are social media platforms that are familiar to many people, and many online users use them every day [43]. Studies [6, 3] explore the growing importance of social networks in contemporary development, investigating the relationship between social media and crime or homicide. Considering Facebook, the authors suggest that the association between Facebook penetration and crime or homicide is overall negative because social media are used essentially for productive ends. However, reference [6] highlights the prominence of a positive relationship between social media in terms of Facebook penetration and terrorism. The direct link between social media and governance dynamics is studied in references [5, 29], while that between social media and corruption is discussed in reference [30]. These studies show the growing importance of information and communication technology and the spread of social media in the daily life of citizens. These relationships also describe on the tourism sector [4].

With regard to user privacy, [48] highlights the fact OSNs' privacy settings and features are frequently concealed, too difficult to understand, or change so quickly that it is nearly impossible for users to maintain them. This complexity raises the question of whether OSN users' privacy settings match what they intend to share. In reference [33], the authors present the results of an empirical evaluation that estimates privacy objectives and behaviors and compares these with the privacy settings on Facebook. This study emphasizes that, although Facebook provides the opportunity to configure privacy preferences at a detailed level on most user data (e.g. each album, video, photo or status update), users have to manage so much data that, even if they used privacy-preserving features regularly, they would not be able to monitor everything. Considering that every participant in this study stated that they had identified at least a sharing violation, the results of this paper show that managing privacy settings is not an easy task both for OSNs and users. A similar study [32] shows results obtained by deploying a survey implemented as a Facebook application. The primary purpose of this study is to quantify the extent of the problem of handling privacy and measuring the discrepancy between the desired and real privacy features. The analysis shows that nearly half of the content users uploading to Facebook

https://doi.org/10.1016/j.ins.2021.01.004

are exposed to all Facebook users as a result of being shared with the default privacy settings.

Another interesting perspective on this topic is given in reference [45], which discusses the privacy risks that come from various recent personalization tendencies. The authors note that, depending on the functionalities offered, each social network handles its users' privacy differently. In reference [50] the authors consider the fact that third parties have control over an enormous amount of personal data and, considering the recent rise in reported privacy violation incidents, they acknowledge that a solution to such problems is possible thanks to verifiable computing achieved using a decentralized network of peers accompanied by a public ledger.

A relevant topic related to the proposed approach is blockchain. In reference [25] the limitations and the advantages of using private blockchain in businesses are explored. The authors propose an attribute-based encryption security system that relies on a private-over-public (PoP) blockchain approach to overcome the drawbacks of private blockchain and simultaneously to benefit fully from the positive features of the public blockchain.

Blockchain and smart contracts are applied to many domains including AI, 5G, IoT, and proof of delivery systems. The study shown in reference [42] suggests how blockchain technology could transform AI, solving many shortcomings and challenges related to AI such as data security, collective decision making, and decentralized intelligence. Investigating on the features of blockchain architecture and platforms, the authors show a wide range of open research challenges for combining AI and blockchain technologies.

The authors of reference [11] discuss the key opportunities offered by blockchain technology in 5G networks, highlighting the challenges of scalability and interoperability among different blockchain platforms and 5G stakeholders. Concurrently, IoT applications are continuously growing and, these devices are deployed at a massive scale. The authors of reference [1] propose a blockchain-based authentication system and implementation relying on Ethereum smart contracts for IoT devices.

With regard to the proof of delivery of assets, references [22, 23] explore the use of the Ethereum blockchain to create decentralized PoD systems ensuring accountability and integrity. Their solution includes the contemporary presence of multiple transporters, eliminating the need for a trusted third party. In reference [24], another blockchain-based solution to show the delivery of digital assets is shown. Their proposal is developed by smart contracts exploiting the IPFS decentralized file system to ensure that the

https://doi.org/10.1016/j.ins.2021.01.004

integrity of the agreement form between the parties is well maintained.

Every day sensitive and behavioral data about users are collected by social networks [13] and present a means for companies or attackers that use them for marketing or other purposes. The authors propose a blockchain-based model to guarantee the privacy of users and to protect sensitive personal information in a distributed environment. This approach focuses on data storage applications, and the DEPLEST algorithm solves the problem of data synchronization integrated with a new consensus protocol for blockchain ledgers.

The investigation of transactions recorded inside blockchains is also important. Reference [21] studies transaction features and the relationships between them. The authors introduce statistical laws of the data related to a framework of network science and they represent the relations between different user accounts as a graph. They believe that this statistical approach can be replicated to other cryptocurrency platforms.

The most recent literature related to the privacy aspects discussed in this paper is described in the following. The authors of reference [41] highlight the importance of self-determining privacy settings by digital service users: specifically, the selected privacy requirements must be correct and describe the real privacy demands of users. The study shows a categorization of the common types of specification privacy interfaces and different user types. The experiments identify how to increase the effectiveness, efficiency, and satisfaction of privacy policy specification interfaces.

In some cases, the privacy settings of many mobile apps are difficult to understand and locate by users [12]. These difficulties expose user privacy to various risks, without proper consent. The authors of reference [12] report a systematic study of this problem and analyze the user perception of privacy settings. They discovered that 82.16% of hidden privacy settings are set to leak user privacy by default. Privacy settings suffer from the "set it and forget it" issue [34]. The decisions made about users' personal preferences could change over time and users tend to forget this type of setting. The survey analyzes the behavior of 78 Facebook users to understand the potential risks of the incorrect shifting of privacy preferences and to identify error-prone and mismatched privacy settings.

Reference [40] describes the basic interest of users in transparency and privacy control measures, individuating two important topics: transparency and self-determination. Privacy settings are generally perceived as too complex, and taking actions and making decisions about privacy are difficult for

https://doi.org/10.1016/j.ins.2021.01.004

many users. Even if users are interested in protecting their privacy, they are frequently hindered from making decisions.

This review of the literature shows the potential of blockchain technology in applications that support privacy and authorization management. The importance of privacy aspects in OSNs, such as users' awareness of validating their privacy choices, is evident. To the best of our knowledge, the unfair behavior of a social network that changes the privacy settings of the user is a new problem and no solution has been proposed in the literature. The proposed approach is the first technique that detects the alteration of the privacy settings of a user performed by a social network, and this result is obtained by exploiting the power of blockchain technology.

7. Discussion

In this section, we discuss certain aspects of the proposed solution.

We start by managing how the goals of the proposed solution are reached. The blockchain stores all events related to the privacy settings of a user, which are when a user states the desired privacy settings and when a social network assigns the privacy settings to a user. In the event of a complaint, a misbehaving party can be detected by looking at the events on the blockchain

The events on the blockchain allow a social network to demonstrate that they have acted honestly, thus providing accountability, because the smart contract verifies that the privacy settings assigned to a user are compliant with her/his expectations.

The blockchain stores the privacy settings of a user. However, no party except the user and the social network can link these settings to the user, because the settings are associated with just a blockchain address, which is pseudo-anonymous [50]. The link is generated in the step *user verification* and is published on the Blockchain as a digest and can be known only by guessing the password of the user, which can be assumed to be a secret only known by the user and the social network.

We include certain considerations related to the cost of implementing this solution by reporting the price of the operations related to the creation of and the calls to the smart contract. The deployment of the smart contract, which is performed only once, costs 525 Micro(ETH) (in December 2020, this is approximately 0.12 $); the call to the function `setPrivacy` costs 91 Micro(ETH) (approximately 0.02 $) and the function `checkSettingGraph` costs 29 Micro(ETH) (approximately 0.0065 $). Thus, we can conclude that

https://doi.org/10.1016/j.ins.2021.01.004

by paying about 2 cents, users can set their privacy settings, and the OSN, with a cost of 0.65 cents per user, can check whether the privacy settings of the users are compliant with their privacy features. We believe that these costs can be borne by the social network, even though there can be applications in which this service is paid by users. This result allows us to state that the implementation of the proposed solution is cheap and effective.

Despite these important results, the proposed solution have certain limitations. As discussed above, the drawback of using Ethereum is related to the transaction fee. We noted in Section 5.1 that other implementations of blockchain could be used, such as IOTA [36] and EOSIO [16], to overcome this problem.

Another limitation of the proposed solution is related to the creation of the privacy feature graph of a social network (see Definition 1). The identification of group and resource nodes requires a study of the social network and cannot be automated. Also, any change in the privacy options of a social network requires updating the graph. Fortunately, these changes are not frequent in social networks but do also require a change in the user interface of the website and app.

A final aspect to consider regards the security of the smart contract code: it is critical to ensure that the smart contract code is bug-free and is not vulnerable to almost any security threats [24, 1]. One of the most effective approaches to create secure applications is to make the implementation open source (as we did) in such a way that many programmers may access and test the code. However, we must remember that it is impossible to make a system completely secure and still usable.

8. Conclusion

The problem of guaranteeing the privacy of users in the context of social networks is receiving increasing attention from the research community. The case of Cambridge Analytica and the issue of the GDPR Regulation have highlighted the need to increase research to discover possible privacy issues and suitable solutions.

This paper provides knowledge to this field of study by highlighting possible misbehavior of a social network that can result in a privacy violation. The privacy settings of a user are stored by the social network, which acts as a privileged party and could modify the user's choices to spread his/her data at any time without a user being able to prove this violation. These aspects

https://doi.org/10.1016/j.ins.2021.01.004

have become critical challenges with the issuance of the GDPR Regulation, and the proposed approach aims to provide OSNs with a tool to demonstrate their honesty. The proposed approach combines the use of blockchain technology with a highly adaptable model to define the privacy settings of users in social networks. The proposed solution has been implemented to show its effectiveness and cheapness: the Ethereum smart contract implementing the required functionality, and the decentralized web application that serves as a user interface to interact with the smart contract. The price for obtaining such features is on the order of some cents for each user.

The change required for a social network is minimal. They should show users the possibility to declare the blockchain address used to manage privacy, which is associated with the decentralized application. Despite this negligible change, social networks would markedly benefit from using the proposed solution.

As future work, we intend to extend the model to allow the same graph to be used for more social networks. We plan to allow a user to select general (i.e., independent from a specific social network) privacy settings, and these settings are then automatically converted to specific privacy settings for each social network of the user.

References

[1] Randa Almadhoun, Maha Kadadha, Maya Alhemeiri, Maryam Alshehhi, and Khaled Salah. A user authentication scheme of iot devices using blockchain-enabled fog nodes. In *2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA)*, pages 1–8. IEEE, 2018.

[2] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, pages 1–15, 2018.

[3] Simplice Asongu, Jacinta Nwachukwu, Stella-Maris Orim, and Chris Pyke. Crime and social media. *Information Technology & People*, 2019.

[4] Simplice Asongu and Nicholas M Odhiambo. Tourism and social media

https://doi.org/10.1016/j.ins.2021.01.004

in the world: An empirical investigation. *Journal of Economic Studies*, 2019.

[5] Simplice A Asongu and Nicholas M Odhiambo. Governance and social media in african countries: An empirical investigation. *Telecommunications Policy*, 43(5):411–425, 2019.

[6] Simplice A Asongu, Joseph I Uduji, and Elda N Okolo-Obasi. Homicide and social media: Global empirical evidence. *Technology in Society*, 59:101188, 2019.

[7] Oshrat Ayalon and Eran Toch. Not even past: Information aging and temporal privacy in online social networks. *Human–Computer Interaction*, 32(2):73–102, 2017.

[8] Charles Baden-Fuller and Stefan Haefliger. Business models and technological innovation. *Long range planning*, 46(6):419–426, 2013.

[9] Andreas Bogner, Mathieu Chanson, and Arne Meeuw. A decentralised sharing app running a smart contract on the ethereum blockchain. In *Proceedings of the 6th International Conference on the Internet of Things*, pages 177–178. ACM, 2016.

[10] Richard Gendal Brown, James Carlyle, Ian Grigg, and Mike Hearn. Corda: an introduction. *R3 CEV, August*, 1:15, 2016.

[11] Abdulla Chaer, Khaled Salah, Claudio Lima, Pratha Pratim Ray, and Tarek Sheltami. Blockchain for 5g: opportunities and challenges. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2019.

[12] Yi Chen, Mingming Zha, Nan Zhang, Dandan Xu, Qianqian Zhao, Xuan Feng, Kan Yuan, Fnu Suya, Yuan Tian, Kai Chen, et al. Demystifying hidden privacy settings in mobile apps. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 570–586. IEEE, 2019.

[13] Yun Chen, Hui Xie, Kun Lv, Shengjun Wei, and Changzhen Hu. Deplest: A blockchain-based privacy-preserving distributed database toward user behaviors in social networks. *Information Sciences*, 2019.

[14] Bhaskar DasGupta, Nasim Mobasheri, and Ismael G Yero. On analyzing and evaluating privacy measures for social networks under active attack. *Information Sciences*, 473:87–100, 2019.

https://doi.org/10.1016/j.ins.2021.01.004

[15] Andreas Ellervee, Raimundas Matulevicius, and Nicolas Mayer. A comprehensive reference model for blockchain-based distributed ledger technology. In *ER Forum/Demos*, pages 306–319, 2017.

[16] Eosio blockchain official website. `https://eos.io/`, 2020.

[17] How can i adjust my facebook privacy settings? `https://www.facebook.com/help/193677450678703`, 2020.

[18] Bogdan Cristian Florea. Blockchain and internet of things data provider for smart applications. In *2018 7th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4. IEEE, 2018.

[19] GDPR official website. `https://eugdpr.org/`, 2019.

[20] Gideon Greenspan. Multichain private blockchain-white paper. *URl: http://www. multichain. com/download/MultiChain-White-Paper. pdf*, 2015.

[21] Dongchao Guo, Jiaqing Dong, and Kai Wang. Graph structure and statistical properties of ethereum transaction relationships. *Information Sciences*, 492:58–71, 2019.

[22] Haya R Hasan and Khaled Salah. Blockchain-based proof of delivery of physical assets with single and multiple transporters. *IEEE Access*, 6:46781–46793, 2018.

[23] Haya R Hasan and Khaled Salah. Blockchain-based solution for proof of delivery of physical assets. In *International Conference on Blockchain*, pages 139–152. Springer, 2018.

[24] Haya R Hasan and Khaled Salah. Proof of delivery of digital assets using blockchain and smart contracts. *IEEE Access*, 6:65439–65448, 2018.

[25] Dijiang Huang, Chun-Jen Chung, Qiuxiang Dong, Jim Luo, and Myong Kang. Building private blockchains over public blockchains (pop) an attribute-based access control approach. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 355–363, 2019.

[26] Yuheng Huang, Haoyu Wang, Lei Wu, Gareth Tyson, Xiapu Luo, Run Zhang, Xuanzhe Liu, Gang Huang, and Xuxian Jiang. Characterizing eosio blockchain. *arXiv preprint arXiv:2002.05369*, 2020.

https://doi.org/10.1016/j.ins.2021.01.004

[27] Instagram privacy settings and information. `https://help.instagram.com/196883487377501`, 2020.

[28] Jim Isaak and Mina J Hanna. User data privacy: Facebook, cambridge analytica, and privacy protection. *Computer*, 51(8):56–59, 2018.

[29] Chandan Kumar Jha and Oasis Kodila-Tedika. Does social media promote democracy? some empirical evidence. *Journal of Policy Modeling*, 2019.

[30] Chandan Kumar Jha and Sudipta Sarangi. Does social media reduce corruption? *Information Economics and Policy*, 39:60–71, 2017.

[31] Ioannis Karamitsos, Maria Papadaki, and Nedaa Baker Al Barghuthi. Design of the blockchain smart contract: A use case for real estate. *Journal of Information Security*, 9(3):177–190, 2018.

[32] Yabing Liu, Krishna P Gummadi, Balachander Krishnamurthy, and Alan Mislove. Analyzing facebook privacy settings: user expectations vs. reality. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 61–70. ACM, 2011.

[33] Michelle Madejski, Maritza Lupe Johnson, and Steven Michael Bellovin. The failure of online social network privacy settings, 2011.

[34] Mainack Mondal, Günce Su Yilmaz, Noah Hirsch, Mohammad Taha Khan, Michael Tang, Christopher Tran, Chris Kanich, Blase Ur, and Elena Zheleva. Moving beyond set-it-and-forget-it privacy settings on social media. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 991–1008, 2019.

[35] Openchain 0.7 documentation. `https://docs.openchain.org/en/latest/`, 2020.

[36] Serguei Popov. The tangle (2016). *Verfügbar: http://www. tangle-blog. com/wpcontent/uploads/2016/11/IOTA_Whitepaper. pdf.[Zugriff am 22.05. 2019]*, 2018.

[37] Lijin Quan, Lei Wu, and Haoyu Wang. Evulhunter: Detecting fake transfer vulnerabilities for eosio's smart contracts at webassembly-level. *arXiv preprint arXiv:1906.10362*, 2019.

https://doi.org/10.1016/j.ins.2021.01.004

[38] Shailendra Rathore, Pradip Kumar Sharma, Vincenzo Loia, Young-Sik Jeong, and Jong Hyuk Park. Social network security: Issues, challenges, threats, and solutions. *Inf. Sci.*, 421:43–69, 2017.

[39] Joel R Reidenberg, Travis Breaux, Lorrie Faith Cranor, Brian French, Amanda Grannis, James T Graves, Fei Liu, Aleecia McDonald, Thomas B Norton, and Rohan Ramanath. Disagreeable privacy policies: Mismatches between meaning and users' understanding. *Berkeley Tech. LJ*, 30:39, 2015.

[40] Manuel Rudolph, Denis Feth, and Svenja Polst. Why users ignore privacy policies–a survey and intention model for explaining user privacy behavior. In *International Conference on Human-Computer Interaction*, pages 587–598. Springer, 2018.

[41] Manuel Rudolph, Svenja Polst, and Joerg Doerr. Enabling users to specify correct privacy requirements. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 39–54. Springer, 2019.

[42] Khaled Salah, M Habib Ur Rehman, Nishara Nizamuddin, and Ala Al-Fuqaha. Blockchain for ai: Review and open research challenges. *IEEE Access*, 7:10127–10149, 2019.

[43] Cass R Sunstein. Valuing facebook. *Behavioural Public Policy*, pages 1–12, 2019.

[44] Take control of your Virtual Identity. `https://ec.europa.eu/commission/sites/beta-political/files/digital_avatar_280519_v5.pdf`, 2019.

[45] Eran Toch, Yang Wang, and Lorrie Faith Cranor. Personalization and privacy: a survey of privacy risks and remedies in personalization-based systems. *User Modeling and User-Adapted Interaction*, 22(1-2):203–220, 2012.

[46] How to protect and unprotect your tweets. `https://help.twitter.com/en/safety-and-security/how-to-make-twitter-private-and-public`, 2020.

https://doi.org/10.1016/j.ins.2021.01.004

[47] Paul Van Schaik, Jurjen Jansen, Joseph Onibokun, Jean Camp, and Petko Kusev. Security and privacy in online social networking: Risk perceptions and precautionary behaviour. *Computers in Human Behavior*, 78:283–297, 2018.

[48] Pamela J Wisniewski, Bart P Knijnenburg, and Heather Richter Lipford. Making privacy personal: Profiling social network users to inform privacy education and nudging. *International Journal of Human-Computer Studies*, 98:95–108, 2017.

[49] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.

[50] Guy Zyskind, Oz Nathan, et al. Decentralizing privacy: Using blockchain to protect personal data. In *2015 IEEE Security and Privacy Workshops*, pages 180–184. IEEE, 2015.

https://doi.org/10.1016/j.ins.2021.01.004

```
pragma solidity ^0.5.10;

contract PrivacyManager {
    struct UserSettingStruct {
        bytes setting_graph;
        uint index;
    }
    mapping(address => UserSettingStruct) private userSettingStructs;
    address[] private userSettingIndex;
    event PrivacySetting(address user, bytes setting);
    event SettingChange(address user, bytes setting);
    event CheckSetting(address OSN, address user, bool result);

    function isSet(address _address) public view returns(bool exists) {
        if (userSettingIndex.length == 0)
            return false;
        return (userSettingIndex[userSettingStructs[_address].index] == _address);
    }
    function setPrivacy(bytes memory _setting_graph) public returns(bool success) {
        userSettingStructs[msg.sender].setting_graph = _setting_graph;
        if (!isSet(msg.sender)) {
            userSettingStructs[msg.sender].index = userSettingIndex.push(msg.sender) - 1;
            emit PrivacySetting(msg.sender, _setting_graph);
            return true;
        }
        else {
            emit SettingChange(msg.sender, _setting_graph);
            return true;
        }
    }
    function checkSettingGraph(address _user, bytes memory _request) public {
        if (!isSet(_user)) {
            emit CheckSetting(msg.sender, _user, false);
            return;
        }
        bytes memory setting_graph = userSettingStructs[_user].setting_graph;
        if (setting_graph.length != _request.length) {
            emit CheckSetting(msg.sender, _user, false);
            return;
        }
        else {
            for (uint i = 0; i < setting_graph.length; i++)
                if ((setting_graph[i] | _request[i]) > setting_graph[i]) {
                    emit CheckSetting(msg.sender, _user, false);
                    return;
                }
            emit CheckSetting(msg.sender, _user, true);
        }
    }
}
```

Figure 7: Code of the smart contract.