Information Systems Frontiers
https://doi.org/10.1007/s10796-021-10150-8

A System to Access Online Services with Minimal Personal Information Disclosure

Antonia Russo[1] · **Gianluca Lax**[1] · **Baptiste Dromard**[2] · **Menad Mezred**[2]

Accepted: 21 May 2021

Abstract
The General Data Protection Regulation highlights the principle of data minimization, which means that only data required to successfully accomplish a given task should be processed. In this paper, we propose a Blockchain-based scheme that allows users to have control over the personal data revealed when accessing a service. The proposed solution does not rely on sophisticated cryptographic primitives, provides mechanisms for revoking the authorization to access a service and for guessing the identity of a user only in cases of need, and is compliant with the recent eIDAS Regulation. We prove that the proposed scheme is secure and reaches the expected goal, and we present an Ethereum-based implementation to show the effectiveness of the proposed solution.

Keywords Blockchain · Smart contracts · eIDAS Regulation · Access control · Privacy

1 Introduction

In the digital era, information is a valuable asset: for instance, think about social networks, which collect information of hundreds of millions of people such as personal data, friends, visited places, listened to music, watched TV, preferences, and interests. Such data are monetized in several ways, such as to produce context-aware information that influences users' preferences to recommend specific items (custom advertising). For this reason, often accesses to services require that a user authenticates.

✉ Gianluca Lax
lax@unirc.it

Antonia Russo
antonia.russo@unirc.it

Baptiste Dromard
dromard@ecole.ensicaen.fr

Menad Mezred
mmezred@ecole.ensicaen.fr

[1] University of Reggio Calabria, Reggio Calabria, Italy

[2] ENSICAEN, Caen, France

However, in many situations, there is not a real need to be aware of personal information, which is done only to collect rich data. Consider the case of a merchant selling products for adults (e.g., liquor or cigarettes), which only needs to check that acquirers are of a certain age: the request of the identity card to show the birth date has the side effect of disclosing personal information, such as name, surname, nationality. Although this situation is not very worrying in real life, the problem is relevant in the digital world because disclosed data can be stored and processed automatically. As collected data may contain private information that could be transferred to unauthorized parties, privacy-preserving proposals in this context are gaining attention (Shin, 2010; Kim et al., 2018). Indeed, recently many service providers require the disclosure of less sensitive information.

This privacy problem is well-known and has been recently remarked also by the issuance of the General Data Protection Regulation (European Parliament, 2016), the new European Union privacy law that puts guidelines and regulations on how data have to be processed, used, stored, or exchanged to protect and ensure individuals data privacy (Lee et al., 2020; Shin et al., 2019). In this context, the problem we address is how to build a system able to reach four research goals:

RG1. when accessing a service, a user should be allowed to provide the minimal amount of personal information needed to access the service;

Published online: 19 July 2021

RG2. only authorized users should access a service;
RG3. there should be the possibility to revoke the permission given to a user to access a service;
RG4. the user's identity should be revealed in case of need.

For this purpose, we exploit the power of Blockchain (Nakamoto, 2008), a recent technology used in many fields, such as finance, smart cities, society progress driving (Shin & Ibahrine, 2020). Blockchain is a fully distributed repository that stores transactions. Several nodes distributed in a peer-to-peer fashion have the control over stored information and run programmable rules in the form of *smart contracts* (Karamitsos et al., 2018). By replacing a single centralized party with a distributed ledger of replicated, shared, and synchronized data, Blockchain guarantees transparency, traceability, and immutability of registered information.

In this paper, we propose a Blockchain-based system that allows users to prove the possession of some attributes without disclosing their identity. Moreover, our proposal provides suitable mechanisms to allow revocation and accountability. Revocation concerns the possibility to make invalid a credential when a user loses possession of an attribute (for example, a driver's license) or the credential is stolen or expired. Accountability is a feature that allows a party in cooperation with other trusted parties to guess the identity of a user in cases of need. Differently from the state of the art, our proposal does not rely on sophisticated cryptographic primitives, which reduce the efficiency of a solution. An important aspect is related to the Regulation (EU) N 910/2014 (European Commission, 2016), which regards electronic identification and trust services for electronic transactions in the EU internal market. This regulation provides a normative basis to enable secure electronic interactions between businesses, citizens, and public authorities. Among others, eIDAS introduces the role of the Attribute Provider, an entity responsible for providing information about electronic identities. The issuance of eIDAS opens the possibility to design new solutions for attribute certification that can be very effective because it is expected that in the next years eIDAS will involve most of EU people. We exploited this opportunity so that the proposed solution is compliant with the eIDAS infrastructure, which increases its effectiveness. We instantiated the general solution to a real-life scenario and described the detailed data workflow to show how our approach can be implemented by Ethereum, the most used Blockchain enabling smart contracts.

The paper is structured as follows. In the next section, we survey the most significant proposals of the state of the art related to the addressed problem. We introduce the background needed to understand the proposal in Section 3. In Section 4, we describe the faced problem. In Section 5, we define the approach used to allow a user to disclose some attributes without revealing personal data. In Section 6, we instantiate the proposed scheme to a specific scenario, and we provide the technical details about how our solution works. The security analysis is presented in Section 7. Finally, in Section 8, we draw our conclusions.

2 Related Work

One of the problems addressed in this paper concerns access control, which ensures that users can access a piece of data only if they are authorized (Goyal et al., 2006). Attribute-Based Access Control (ABAC) is defined as logical access control methodology where authorization to perform a set of operations is determined by evaluating attributes associated with the subject, object, requested operations, and, in some cases, environment conditions against policy, rules, or relationships that describe the allowable operations for a given set of attributes. Hu et al. (2013) define ABAC to understand the real applications of this mechanism. Attribute-Based Access Control is analyzed in real use cases to improve scalability, feasibility, and performances of applications in which the information sharing within and between organizations is expected.

Sahai and Waters (2005) propose the concept of Attribute-Based Encryption. Authors provide an original type of Identity-based Encryption in which the identity consists of an attribute set. Users, with an identity and their attributes, can decrypt a ciphertext encrypted with the same attributes.

Goyal et al. (2006) develop a new cryptographic system for fine-grained access control of shared encrypted data, called Key-Policy Attribute-Based Encryption (KPABE). In this system, ciphertexts are matched with sets of attributes, and private keys are associated with access structures.

The concept of Ciphertext-Policy Attribute-based Encryption is formalized by Bethencourt et al. (2007). In this solution, the policy is associated with the ciphertext and the attributes with the key. A user can decrypt a ciphertext if the user's attributes pass through the ciphertext's access structure.

Attrapadung and Imai (2009) propose a solution combining CP-ABE with KP-ABE. The authors consider a scheme in which both policy and attributes are associated with the ciphertext and key. The attributes are related to the ciphertext, and the policy designs the users who can decrypt. The policy states the kind of ciphertext the user can decrypt.

The Attribute-based proxy re-encryption scheme (ABPRE) (Liang et al., 2009) extends the ABE scheme empowering users with delegating capability in the access control environment. A proxy can be chosen by users to re-encrypt a ciphertext related to a specific access policy.

Concerning the revocation of attributes, Hur and Noh (2010) propose a solution exploiting ciphertext-policy attribute-based encryption for an access control mechanism. The solution is related to an efficient implementation provided with an attribute and user revocation capability.

Attribute-based encryption has always been considered as a technology for solving the problem of data privacy and fine-grained access control in traditional cloud storage systems based on a centralized storage architecture. The development of Blockchain technology allows the building of a decentralized storage mode that could overcome the problem of a single point of failure in traditional cloud storage. Wang et al. (2018) propose a framework that combines the Ethereum Blockchain and ABE technology to implement data storage and sharing scheme for decentralized storage systems.

Maesa et al. (2017) deal with a new approach to access control based on Blockchain technology. The policies that express the right to access a resource are published inside Blockchain. That way, every user can check if policies and resources match. Considering a Blockchain, its capabilities of transparency and immutability allow a distributed consensus and auditability preventing a party from denying the rights granted by the policy.

Blockchain technology can provide patients with immutable records regarding their medical data, said Electronic Health Records (EHRs). Guo et al. (2018) present an attribute-based signature scheme with various authorities to enforce the validity of EHRs stored in a Blockchain. This system allows patients to possess the control of generating, managing, and sharing EHRs with other authorized data consumers in a secure environment.

In a cloud computing environment, service providers can be allowed to take care of confidential data, and this permission may raise potential security and privacy issues (Wang et al., 2010). The cloud service provided, by adopting an encryption system, has to support fine-grained access control and also provide high performance and scalability. The authors propose a scheme to help companies use cloud servers to share confidential data efficiently.

Pinno et al. (2017) provide a new architecture for access control in the IoT. This framework, based on Blockchain technology, overcomes the FairAccess (Ouaddah et al., 2016) and other issues derived from the architecture, evaluating a new decentralized and authorization process for authorization in IoT environments. FairAccess is an authorization management framework that is fully decentralized and privacy preserving and uses Blockchain as a decentralized access control manager. With the help of Blockchain, this solution introduces various and new types of transactions to delegate, revoke, and grant access.

From the review of the state of the art here reported, it emerges the importance of the attribute-based access control problem, which is faced by various sophisticated cryptographic approaches. The first observation is that most of the proposals in the field of selective disclosure (for example, Hernández-Ramos et al. (2018) and Lorünser et al. (2016)) rely on sophisticated cryptographic primitives. In contrast, our proposal relies only on efficient and straightforward cryptographic operations, which are considered secure and efficient. Moreover, our proposal is designed for taking advantage of the attribute providers defined in eIDAS. To the best of our knowledge, our proposal is the first one compliant with the eIDAS Regulation, which will be a de-facto standard in the next years (at least, in the European Union).

3 Blockchain and eIDAS

In this section, we recall some concepts used in this paper, which are Blockchain technology and eIDAS ecosystem.

Blockchain is a technology that could have the capacity and potency for enhancing and changing various aspects of economy and society, and for this reason, it can be considered as a disruptive technology (Swan, 2015). It was proposed by Nakamoto (2008), and it is defined as a distributed ledger that stores, in a transparent and immutable way, transactions executed among users. Information is stored inside blocks, whose number and dimensions are continuously growing. Every block is linked to the chain by its header. The header contains the hash of the previous block and a timestamp. The transactions that take place in Blockchain are stored inside blocks and contain information on the recipient's public address, the characteristics of the transaction, and the cryptographic signature, which guarantees the integrity and authenticity of the transaction. Every operation has to be confirmed and validated by Blockchain participants, and this concept is summed up by distributed consensus. This way, users can trust the system of the public ledger, without trusting a central authority or a third-party intermediary. It has been proved that perceived privacy in using blockchain positively could affect and influence users' trust and attitudes towards blockchain technology (Shin, 2019; Shin & Hwang, 2020). Over time, Blockchain assumed different meanings and definitions: the first one is called Blockchain 1.0, and it is referred to as the Bitcoin paradigm. This kind of system represents a platform in which it is possible running and deploying all the operations carried out with cryptocurrency in digital payment systems.

Ethereum (2020) is a Blockchain 2.0 that enables the possibility to create and run *smart contracts*, programs executed over the Ethereum computing infrastructure. A smart contract is defined as a digital protocol to verify or digitally enforce the negotiation of a contract, with no need for an intermediary. The code of a smart contract

automatically verifies the occurrence of specific conditions and automatically carries out actions when the conditions determined between the parties are reached. Smart contracts are written in Solidity (2020), an object-oriented and high-level language, and a real application relies on the creation and the deployment of distributed applications (Ethereum dApps, 2020). It exists a practical and conceptual issue about the external data used to verify and perform decision inside smart contracts, that is the "oracle" presence in Ethereum. An oracle has the purpose of connecting decentralized applications with third-party services in a trust and secure way, in order to get data from outside Blockchain and execute any API call preventing data integrity and authenticity. Provable blockchain (2020) is the most famous oracle service for smart contracts and Blockchain applications. It consists of three main entities: data-source, query, and oracle. When a smart contract requires data from a data source outside Blockchain, it sends a query to Provable and calls a function passing the result of the query as an input. Provable aims to demonstrate that the data taken from the original data source is authentic, by linking returned data with an authenticity proof document.

The Regulation (EU) N 910/2014 (European Commission, 2016) on electronic identification and trust services for electronic transactions in the internal market provides a normative basis to enable secure electronic interactions between businesses, citizens, and public authorities. The eIDAS principles are based on security, trust, and interoperability of electronic services carried out by citizens all over EU countries. All Member States have to notify their eID schemes (national electronic identification schemes) to the European Commission, which are published in the Official Journal of the European Union. Both people and companies can access public services provided by an EU Member State using the eID of another EU Member State: this concept aims at promoting cooperation between states. Interoperability between different eID-schemes is reached by defining the interfaces between eIDAS-Nodes. However, the eID ecosystem consists of various actors that should be available in all EU countries: the most important is the node operator, which controls that an eID node behaves correctly and implements the function of the connection point between the attribute provider, the identity provider and the service provider.

In our proposal, we will focus on the role of attribute providers and, in particular, those compliant with eIDAS Regulation. An attribute provider is in charge of providing information about electronic/digital identities: specifically, this entity has to verify the real possession of different attributes from a digital identity. A second entity, the identity provider, creates and manages the public digital identity of a physic or legal individual. Instead, a service provider is a public or private entity that makes available online services using eID for cross-border authentication. Inside the eIDAS framework, eIDAS-Connector and eIDAS-Service need to exchange messages regarding personal and technical attributes to support cross-border identification and authentication processes. To do this, they use SAML (2020).

SAML (Security Assertion Markup Language) is an XML-based open-source framework created for the exchange of authentication and authorization information between the service provider and the identity or attribute provider in a secure way. The SAML 2.0 specification defines a series of request/response protocols and assertions that allow an application to request or query an assertion or to ask a user for authentication. Regarding the proof of attribute possession during an authentication process, the complete list of attributes included in the eIDAS minimum data sets is declared in the technical document (CEF Digital, 2019) concerning eIDAS SAML Attribute Profile.

4 Scenario and Problem Formulation

In this section, we introduce a scenario to present the addressed problem.

The scenario is composed of the following actors:

- *Users*, physical or legal people with an eIDAS-compliant digital identity.
- *Identity Providers*, eIDAS-compliant entities that create and manage digital identities.
- *Attribute Providers*, eIDAS-compliant entities that are in charge of verifying and validating the possession of attributes.
- *Service Providers*, which supply users with (online) services.
- a *Blockchain*, a Distributed Ledger in which smart contracts can be deployed.

We consider the case of a user who needs to access an online service supplied by a service provider, but this access is granted, provided that the user has the permissions and attributes to access it. The considered service is a car rental booking, in which users need to demonstrate the possession of two attributes: a driver's license and the age (because the rental price depends on the user's age). Thus, a service provider that offers a rental car service needs to be sure about the possession of the driver's license and to know the age of any user who wants to rent a car. Generally, in such cases, a service provider asks users to provide all personal data, which characterize them as legal and natural people. In some cases, the identification can be performed with the support of a third party (such as a social network authentication procedure).

The problem is that users have to reveal many personal and (possible) sensitive data that are not useful to gain the service requested, and this could damage their privacy and expose them to various attacks or future vulnerabilities. The disclosure of specific attributes can help to reduce this effect: users are responsible for the information they want to share with third parties, that is, they can decide to show data in a granular way.

In summary, the goals to reach are:

1. the service provider should be aware of the minimal amount of the user's information needed to access the service;
2. only authorized users should access a service;
3. there should be the possibility to revoke the permission given to a user to access a service;
4. the user's identity should be revealed in case of need.

Concerning the last item, it is worth noting that the recent definition of the eIDAS environment can help to face this problem with new tools that did not exist until a few years ago (Priesnitz Filho et al., 2019). Indeed, the identity providers defined by eIDAS keep a series of information related to the digital identity of users, called unqualified or elementary attributes. Furthermore, the attribute providers defined by eIDAS are authorized to certify a qualification and can add other attributes to the digital identity of the user.

As a consequence, we propose a solution based on the disclosure of the attributes selected by users who want to access the service. Besides eIDAS, our proposal also relies on the recent technology of Blockchain, which allows us to design a decentralized approach to maintain a hidden link between users and their attributes. Blockchain (see Section 3) is in charge of guaranteeing transparency and immutability of actions (or transactions) and allowing stakeholders to perform and verify a secure access control relying on attributes through smart contracts.

5 Conceptual Model

Starting from the scenario described in the previous section, here we describe the solution designed to solve the considered privacy issue. The entities in our scenario cooperate by performing the tasks described in the following.

1. *Digital Identity issuing*. Any user running our solution needs to have an eIDAS digital identity. To do this, the user needs to register to one of the available identity providers, which is responsible for the verification of the user identity before issuing the credential of the digital identity.

   The identity provider of a user knows a list of elementary attributes (e.g., date of birth) of the user. On the other side, there are also attribute providers that manage other and not elementary attributes related to this identity (for example, a Motor Vehicle Office plays the role of attribute provider for a driver's license).
2. *Blockchain registration*. In this proposal, users will be referred by their Ethereum address, so that any user has to create an *external owned account*, characterized by a public address and controlled by a private key. The public key derives from the private one and is computed by a cryptographic function of type *elliptic curve point multiplication*. Typically, the Blockchain address is derived from the public key by a cryptographic hash function. For example, in Bitcoin, a user with public key K has an address computed as RIPEMD160(SHA256(K)), where SHA-2 (2020) is a cryptographic hash developed by National Security Agency (NSA) and returns a 256-bit digest, whereas (RIPEMD160, 2020) is a cryptographic hash designed in the open academic community and returns a 160-bit digest (i.e., the address).
3. *Credential issuing*. This operation is carried out by the user when a credential for one or more attributes is needed and involves an attribute provider (say AP[1]). According to the eIDAS protocol, AP acts as a service provider and needs to identify the user by an eIDAS-compliant scheme, as illustrated in Fig. 1.

   First of all, the user sends a request for a credential to AP (Step 1). Then, AP replies with an authentication request to be forwarded to the *Identity Provider* (Step 2). The *Identity Provider* performs a challenge-response authentication with the user (Steps 3 and 4). In case of successful user authentication, the *Identity Provider* prepares the assertion of user authentication, which is forwarded to AP (Step 6). This way, AP is aware of the user's digital identity and can verify if the user owns the required attributes. In this case, AP performs a second task to know the user's Blockchain address (Step 7): AP sends a random string to the user and waits for receiving a Blockchain transaction with this string from the address of the user, say A.

   If such a transaction is received, then AP replies to the user with the requested credential: it consists of an assertion reporting the Blockchain address A of the user, the verified attributes, and a (suitable) URL belonging to the Web domain of the attribute provider (e.g., http://www.attributeprovider.com/8f7b19f38f4c4b10b52de9727e9f0538). Finally, the

[1]For the sake of simplicity, let us assume only one attribute provider handles the needed attributes: in case more attribute providers are involved, this operation is repeated for each of them.

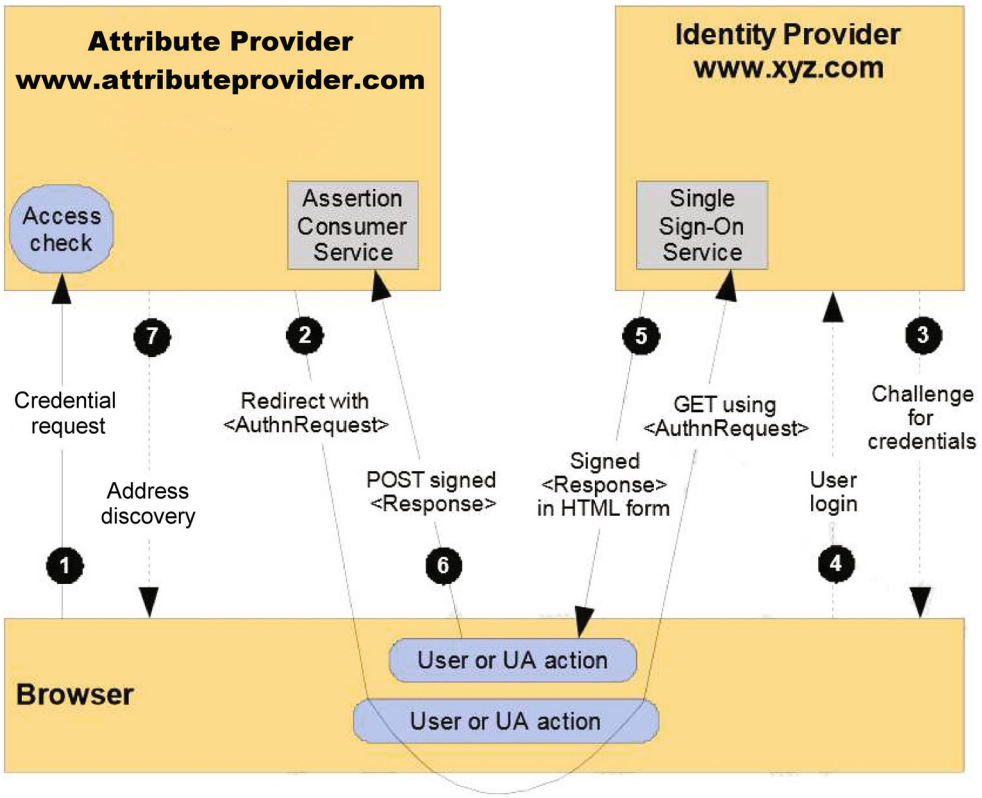


Fig. 1 Data flow in assertion issuing process

attribute provider publishes at this URL the digest of the credential as a proof of authenticity and integrity of the credential (this replaces a cryptographic signature).

4. *Credential using*. When a user needs to access a service granted only to people with some attributes (e.g., a driver's license or being of age), the user can exploit a credential for such attributes obtained with the procedure described in the previous step. The user sends the credential assessing the possession of the requested attributes to the service provider (say SP) supplying this service. The verification of the credential validity is carried out from SP by the call to a function of the smart contract SC, which receives this credential and executes the following steps: (1) extracts from the credential the value of the URL field, (2) verifies that this URL belongs to the domain of the attribute provider, (3) downloads the file at this URL, (4) calculates the digest of this file, say D, and (5) calculates the digest of the received credential, say D'. If and only if $D = D'$, then the function returns that the credential is valid. Only in this case, the user will be able to access the service; otherwise, the access is denied.
5. *Credential Revocation*. Revocation concerns the possibility to make invalid a credential when a user loses possession of an attribute (for example, a driver's license) or the credential is stolen or expired. Revocation is carried out by the attribute provider: for each credential to be revoked, the attribute provider extracts the specified URL where the credential is published and removes the document on this URL, making the file unreachable. This way, the user cannot provide the proof of the requested attribute to any service provider, which cannot find the credential.

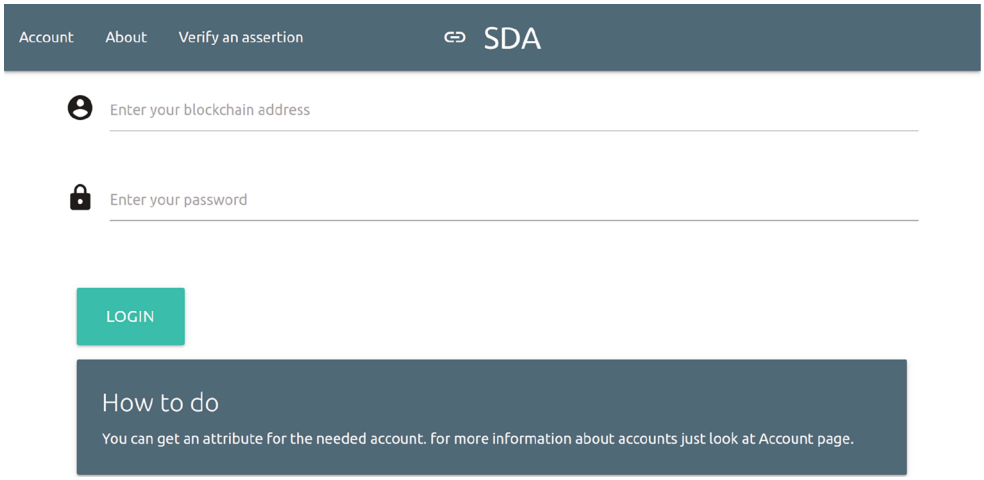
The proposal here presented is defined at a conceptual level and does not consider several (orthogonal) aspects, such as the smart contract and the exchanged data. These aspects are the subject of the next section.

6 Implementation and Proof of Concept

In this section, we instantiate the general approach presented above to a real-life scenario to help the reader to understand better how our solution can work. We describe the detailed data workflow to show how our approach can be implemented in a real Blockchain: we used Ethereum, which is the most used one enabling smart contracts.

For the sake of presentation, we refer to a simplified scenario in which the service to access requires the user to be of age (for example, in the case of age-restricted

Fig. 2 User authentication



videos, which are not visible to users who are under 18 years of age). Our implementation is based on the Ethereum Blockchain, and the environment in which we tested our solution is the Ropsten testnet Blockchain, a free Blockchain based on Ethereum using proof of work (Ropsten Testnet Explorer, 2020). The smart contract is build using (Solidity, 2020) and exploits (Provable blockchain, 2020) to import data from external sources (see Section 3). We implemented a JAVA decentralized web application (DAPP) by the Web3j SDK (2020) library and used Infura (Ethereum and IPFS APIs, 2020) as Blockchain infrastructure to access the Ethereum Blockchain.

Now, we describe how the operations defined in our proposal are implemented in this scenario.

1. *Digital Identity issuing*. The public digital identity of a user U can be issued in order to perform a secure authentication based on attributes. The Identity Provider IP stores a list of the elementary attributes of the user. The Attribute Provider AP manages not elementary attributes and checks the identity of the user.
2. *Blockchain registration*. We created an Ethereum address for any entity involved (user, identity provider, and attribute provider). First, we generated a couple of asymmetric keys for each of them. Then, the Ethereum address is computed from the public key by applying Keccak-256 (Bertoni et al., 2009) and taking the last 20 bytes of that hash. In practice, we installed the *MetaMask* extension in Google Chrome, created the new accounts, and saved the seed words for restoring the MetaMask accounts.
3. *Assertion issuing*. This operation is carried out by the user when a proof of attribute is needed.

   To obtain the assertion, the user connects to the AP's site by a browser and sends the request for an assertion. Now, the user is authenticated by the chosen eID (this part is skipped in our implementation because it is a standard procedure). Then, AP replies with a challenge-response authentication with the user (see Fig. 2). In case of successful user authentication, the attribute provider shows a page with all the user's attributes so that the user can select the ones to be certified, as shown in Fig. 3.

   Now, AP prepares a response that includes the assertion of the attributes selected by the user: this message is based on the SAML format and according to eIDAS eID Technical Subgroup (2019) the most relevant fields are:

   - an attribute `Version` with the version of SAML used in the message;
   - an element `StatusCode` with the outcome of the request;
   - an attribute `ID` containing the assertion identifier;
   - an attribute `IssueInstant`, which specifies the instant at which the assertion is issued;
   - an element `Issuer`, which refers to the issuer of the message;
   - an element `AuthnInstant`, which specifies the instant at which the authentication has been performed.
   - an element `AuthnContextClassRef`, which specifies the used authentication method based on a particular class reference.

   An example of a response message is shown in Fig. 4. The status code contains the URL used by AP to publish the assertion. The assertion contains the version, the ID, the issue instant, the reference to the issuer, the Blockchain address of the subject, the end of validity, and the attribute name. Moreover, the assertion has temporal data regarding the start of validity instant, the timestamp of the authentication, and the type of certified attributes (proof of age).

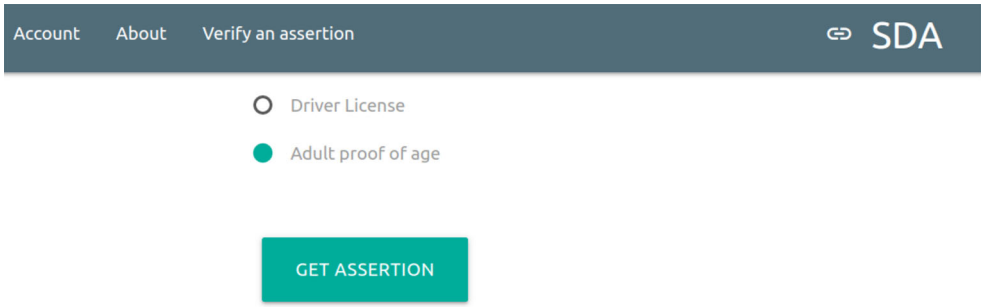


Fig. 3 Attribute selection

Moreover, AP publishes at the URL reported in the assertion a file containing the assertion and calculates the digest of this file. Finally, AP calls the function *indexDigest* of the smart contract (see Fig. 6, Lines 54-59), which generates a token. This token is the digest of a nonce generated from the block timestamp and incremented by one every time a new token is generated. This ensures its uniqueness. Observe that it is not a secret value (all data in the smart contract are not secret). This token is used as an index to find such an assertion digest in a mapping (Line 57).

Finally, the attribute provider sends to the user a JSON file containing the URL where the assertion is available, and the token t. An example of this file is reported in Fig. 5.

4. *Assertion using*. Suppose the user needs a service supplied by the service provider SP and U is required to possess an attribute, for example, a proof of age. To prove to be of age, U sends the JSON file received by the attribute provider in the previous step to SP.

To verify this credential, SP calls the function *verifyAssertion* of the smart contract SC (Fig. 6, Lines 38-49), giving as input the `token` and the `location` extracted from the JSON file. This function is payable (i.e., it can receive Ether) and exploits the Provable oracle to download the assertion located at the given URL (Line 44). Then, it retrieves the digest of the assertion previously stored in *digestSet* (Line 57) and adds it to *verifiedDigest*, a mapping between digests and query identification (i.e., *content* in Line 45). Moreover, the query is included in a set of pending query (line 46).

Once Provable returns the query result, the *callback* function is called automatically. This function has the oracle query identification Id and the query result as parameters. First, the digest of the assertion downloaded by Provable is calculated (Line 29)

Fig. 4 Example of response message

```
<saml2p:Response
    xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" Version="2.0">
    <saml2p:Status>
        <saml2p:StatusCode>
            https://attributeProvider.com/zkikl0NcxdxaUIoqCZfX.xml
        </saml2p:StatusCode>
    </saml2p:Status>
    <saml2:Assertion
        xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion"
            ID="47fd4a3a-16ea-415a-9b16-f9f034785388"
            IssueInstant="2020-01-22T09:19:29.170Z" Version="2.0">
        <saml2:Issuer>attributeProvider.com</saml2:Issuer>
        <saml2:Subject>
            <saml2:NameID
                Format="urn:oasis:names:tc:SAML:2.0:nameid-format:
                    persistent">
                0x8fa7173202d86C746bd884C9f116E356600c6b0E
            </saml2:NameID>
            <saml2:SubjectConfirmation
                Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
                <saml2:SubjectConfirmationData
                    NotOnOrAfter="2020-07-21T09:19:29.163Z" />
            </saml2:SubjectConfirmation>
        </saml2:Subject>
        <saml2:AuthnStatement
                AuthnInstant="2020-01-22T09:19:27.138Z">
            <saml2:AuthnContext>
                <saml2:AuthnContextClassRef>
                urn:oasis:names:tc:SAML:2.0:ac:classes: proof of age
                </saml2:AuthnContextClassRef>
            </saml2:AuthnContext>
        </saml2:AuthnStatement>
    </saml2:Assertion>
</saml2p:Response>
```

Fig. 5 Example of JSON credential

```
{
    "location":"https://attributeProvider.com/zkikl0NcxdxaUIoqCZfX
        .xml",
    "token":"4ce9b145974bad0dbeca705f89d3f44161fc8226348497cd67854
         b0edcf7e465"
}
```

and compared to the digest previously stored in *verifiedDigest* (Line 30). The result of this comparison is stored in *resultSet*.

Then, the Service provider SP calls the function *checkResult*, which takes the query identification Id as a parameter. This function returns the content of the *resultSet* at the index Id, which represents the result of the previous function (Lines 50-53). Specifically, the value 1 denotes a valid assertion, the value 2 denotes an incorrect assertion, whereas the value 0 denotes that the query result is not available (for example, in case of wrong id of the query).

Concerning the smart contract, we showed a more extensive implementation in which also events are used (see Lines 11-17 and the calls to function *emit*). Logged events can provide support in case of need (for example, in the event of a complaint). The implementation of this solution by a Java prototype is available on Github at the address https://github.com/DroBaptiste/SelectiveDisclosureOfAttribute. Concerning the smart contract, it has been deployed on Ropsten and is reported at https://ropsten.etherscan.io/address/0x2cF05A44F23A92581088c17e7C8c7D88B2F8d0f2#code, where the `provable.sol` library has been included.

7 Security Analysis

In this section, we discuss how our proposal reaches the expected goals, which are:

1. the service provider should be aware of the minimal amount of the user's information needed to access the service;
2. only authorized users should access a service;
3. there should be the possibility to revoke the permission given to a user to access a service;
4. the user's identity should be revealed in case of need.

Let us start with the first property. The service provider is aware of (i) the JSON file and (ii) the assertion. The former does not contain any information about the user; the latter contains only data that the user selected. Provided that (1) there is no collusion among identity provider, attribute provider, and service provider, and (2) the user does not select more information than the needed one to access the service, the first goal is reached.

Concerning the second item, it is clear that authorized users can access the service by following the protocol. Thus, we have to prove that unauthorized users cannot create a valid assertion. In our analysis, we assume that the protocol is correctly run by identity and attribute providers because they are trusted entities. Consequently, an attacker cannot tamper with his/her identity or attributes, which are guaranteed by identity and attribute providers. When the attacker is able to create an assertion that passes all the security checks, we have to consider the following two possibilities. The attacker has created a false assertion and is able to create the signature done by the attribute provider. This means to break the cryptographic primitives or to guess the private key. The probability of any of these cases is negligible. Another possibility is that the attacker has violated the smart contract. If we assume no error exists in the smart contract code, then the adversary can only try a 51% attack (Sayeed & Marco-Gisbert, 2019). Again, the probability of this attack is negligible.

Concerning the attribute revocation, it can occur for several reasons: 1) most of the attributes have an expiring date from which their validity ends, 2) a user can choose to change the status of a single attribute (in our scenario attributes could be separate and disjointed) or 3) the most significant cause is when the attribute provider ought to revoke the expired attribute. How the revocation occurs is evident in the protocol. Thus, we focus on a possible misconduct of attribute providers, that is the unfair revocation of an attribute. Consider the case of an attribute provider AP that has issued an adult proof of age credential to a user U. Now assume that AP revokes this credential unfairly so that the credential validation fails. However, if U has stored this credential, then by computing the digest d of the credential and by invoking the function *indexDigest* of the smart contract (see Fig. 6, Lines 54-59), U can prove that a token is associated with d, and thus, that AP had issued this credential. This possible malicious behavior of AP is detected and, thus, contrasted. Clearly, in some cases, other information could be necessary to close the complaint.

As for the last goal, we reach this result by exploiting an eIDAS-compliant identification scheme. The robustness of an identification scheme depends on the degree to which it adheres to the technical specifications and best practices. Even if the standards used for identification systems can vary by country, the compliance with eIDAS ensures acceptable robustness. Our scheme allows a party in cooperation with other trusted parties to guess the identity

Fig. 6 Code of smart contract

```
pragma solidity >= 0.5.0 < 0.6.0;
import "./provable.sol";

contract verificationContract is usingProvable {
mapping (bytes32 => bytes32) private digestSet;
mapping (bytes32 => bytes32) private verifiedDigest;
mapping (bytes32 => bool) private pendingQueries;
mapping (bytes32 => uint) private resultSet;
uint nonce;

event LogConstructorInitiated(string nextStep);
event LogVerification(string saml);
event LogNewProvableQuery(string description);
event AssertionResult(uint answer);
event TokenIndexed(bytes32 token);
event queryInitiated(bytes32 id);
event TransactionMade();

constructor() public payable {
    nonce = block.timestamp;
    emit LogConstructorInitiated("Constructor was initiated");
}
function pay() payable public {
    emit TransactionMade();
}
function __callback(bytes32 myid, string memory result) public{
    if (msg.sender != provable_cbAddress())
        revert();
    require (pendingQueries[myid] == true);
    bytes32 digest = sha256(abi.encodePacked(result));
    if (verifiedDigest[myid] ==  digest) {
        resultSet[myid] = 1;
    } else {
        resultSet[myid] = 2;
    }
    emit LogVerification(result);
    delete pendingQueries[myid];
}
function verifyAssertion(bytes32 _token, string memory _url)
    public payable{
    if (provable_getPrice("URL") > address(this).balance) {
        emit LogNewProvableQuery("Oraclize query was NOT sent,
            please add some ETH to cover for the query fee");
    } else {
        string memory str = strConcat("binary(",_url,").slice
            (0,10000)");
        emit LogNewProvableQuery("Oraclize query was sent,
            standing by for the answer..");
        bytes32 content = provable_query("URL", str ,300000);
        verifiedDigest[content] = digestSet[_token];
        pendingQueries[content] = true;
          emit queryInitiated(content);
    }
}
function checkResult(bytes32 id) public returns(uint) {
    emit AssertionResult(resultSet[id]);
    return resultSet[id];
}
function indexDigest(bytes32  _digest) public{
    bytes32 token = sha256(abi.encodePacked(nonce));
    nonce++;
    digestSet[token] = _digest;
    emit TokenIndexed(token);
}
}
```

of a user in cases of need. This possibility is given because each credential has an identifier, and each attribute provider stores the mapping between each generated credential and the user identity. Recall that a service provider is not aware of the identity of the user accessing the service, and the only information known is about the user's attributes certified by the credential. However, in case of valid reasons (for instance, a terrorist who rented a car), the cooperation

between the service provider (which knows the credential identifier) and the attribute provider that issued such a credential (which knows the user's identity associated with this credential) allows us to uncover the user's identity.

Concerning this aspect, we observe that there is the possibility of running off-line the scheme by relaxing some of the security requirements. Consider the case in which the URL is not available: the protocol should deny the request of access request because the credential cannot be retrieved. However, as it is done for micro-payments when the economic value of the service is limited, the service provider could accept to receive the credential from the user instead of downloading it from the (unavailable) URL. Observe that in this case, the credential verification is done again by the function *indexDigest*, thus proving that a token is associated with this credential. Clearly, in this way, the check of credential revocation has not been carried out: however, if the cost of making unavailable the web site of the attribute provider (for example, by a Denial-of-Service attack) is higher than the price of the service, it is not advantageous for a malicious user to run this attack for obtaining the service. The choice about providing the service at this risk is left to the service provider.

8 Conclusion

In this paper, we proposed a system that allows users to prove the possession of some attributes without disclosing their whole identity. Our solution relies on Blockchain, which is a very recent technology but considered mature and already widely adopted in many application contexts. The use of a Blockchain platform such as Ethereum allowed us to make transparent cryptographic operations to user and system, thus making the solution more effective, robust, and secure. Using certain features of Blockchain, the proposed system provides suitable mechanisms for revocation and accountability. The proposed solution has been implemented by a JAVA decentralized Web application that exploits Ethereum as Blockchain. As a real scenario for validation, we considered an infrastructure compliant with the eIDAS Regulation, in which the attribute providers defined by eIDAS are authorized to certify a qualification or some attributes of a digital identity.

Most of the proposals of the state of the art rely on sophisticated cryptographic primitives. In contrast, our proposal relies only on efficient and straightforward cryptographic operations, which are based on elliptic curves and considered secure and efficient.

The proposed solution has practical implications, primarily related to the General Data Protection Regulation (European Parliament, 2016), which remarks the importance of seven basic principles of data protection: 1) lawfulness, fairness, and transparency, 2) purpose limitation, 3) data minimization, 4) accuracy, 5) storage limitation, 6) security, 7) accountability. Although how to apply these principles is not stated, they represent the spirit of the regulatory framework. Thus, compliance with these principles is fundamental to build any data-processing framework in practice. Indeed, this regulation emphasizes the importance of applying these principles to any company, and it is not possible to be GDPR-compliant without implementing these rules in the data life cycle of the company. The third principle regards data minimization (Article 5.1.c) and requires entities to process only *adequate, relevant, and limited personal data that is necessary*. This regulation does not define what the terms adequate, relevant, and limited means but states that data processing should only use as much data as is required to successfully accomplish a given task, and data collected for one purpose cannot be used for a different purpose without obtaining a new consent. This means that companies must limit personal data collection to data that are absolutely necessary for carrying out the purpose for which data are processed.

In this paper, we described an example in which a company needs to know if a user is of age but collects the user's date of birth. This data processing violates the minimization principle and, according to Article 83 of European Parliament (2016), this infringement is subject to administrative fines up to 20M EUR or, in the case of an undertaking, up to 4% of the total worldwide annual turnover of the preceding financial year.

This aspect highlights the practical importance of our solution: the scheme we proposed allows a company to comply with the data minimization principle stated by GDPR, yet ensuring that access-control policies are respected. Although the use of our solution does not allow a company to know the identity of users accessing a service, in case of valid reasons (for instance, a terrorist who rented a car), it is possible to uncover user's identity with the support of a trusted party, which is the attribute provider that certified user's age. This is an important added value of our proposal, especially in this period in which the balance between privacy right and security right is difficult to determine.

Funding Open access funding provided by Università degli Studi Mediterranea di Reggio Calabria within the CRUI-CARE Agreement.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

Attrapadung, N., & Imai, H. (2009). Dual-policy attribute based encryption. In M. Abdalla, D. Pointcheval, P. A. Fouque, & D. Vergnaud (Eds.) *Applied cryptography and network security* (pp. 168-185). Berlin: Springer.

Bertoni, G., Daemen, J., Peeters, M., & Van Assche, G (2009). Keccak specifications. Submission to nist (round 2), pp 320–337.

Bethencourt, J., Sahai, A., & Waters, B (2007). Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy (SP '07)* (pp. 321–334). https://doi.org/10.1109/SP.2007.11.

CEF Digital (2019). eID Profile. https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eIDAS+eID+Profile, Accessed 13 January 2021.

eIDAS eID Technical Subgroup (2019). eIDAS SAML Message Format. https://ec.europa.eu/cefdigital/wiki/download/attachments/82773108/eIDAS%20SAML%20Attribute%20Profile%20v1.2%20Final.pdf, Accessed 13 January 2021.

Ethereum (2020). https://www.ethereum.org, Accessed 13 January 2021.

Ethereum and IPFS APIs (2020). https://infura.io/, Accessed 13 January 2021.

Ethereum dApps (2020). Explore Decentralized Applications. https://www.stateofthedapps.com, Accessed 13 January 2021.

European Commission (2016). eIDAS Regulation (Regulation (EU) N 910/2014). https://ec.europa.eu/futurium/en/content/eidas-regulation-regulation-eu-ndeg9102014, Accessed 13 January 2021.

European Parliament (2016). General data protection regulation. https://eur-lex.europa.eu/eli/reg/2016/679, Accessed 13 January 2021.

Goyal, V., Pandey, O., Sahai, A., & Waters, B (2006). Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security* (pp. 89–98). ACM.

Guo, R., Shi, H., Zhao, Q., & Zheng, D (2018). Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems. *IEEE Access*, *6*, 11676–11686.

Hernández-Ramos, J.L., Pérez, S., Hennebert, C., Bernabé, J.B., Denis, B., Macabies, A., & Skarmeta, A F (2018). Protecting personal data in IoT platform scenarios through encryption-based selective disclosure. *Computer Communications*, *130*, 20–37.

Hu, V.C., Ferraiolo, D., Kuhn, R., Friedman, A.R., Lang, A.J., Cogdell, M.M., Schnitzer, A., Sandlin, K., Miller, R., Scarfone, K., & et al. (2013). Guide to attribute based access control (abac) definition and considerations (draft). *NIST special publication, 800*(162).

Hur, J., & Noh, D.K. (2010). Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Transactions on Parallel and Distributed Systems*, *22*(7), 1214–1221.

Karamitsos, I., Papadaki, M., & Al Barghuthi, NB (2018). Design of the blockchain smart contract: a use case for real estate. *Journal of Information Security*, *9*(3), 177–190.

Kim, J., Baskerville, R.L., & Ding, Y (2018). Breaking the privacy kill chain: Protecting individual and group privacy online. *Information Systems Frontiers*, 1–15.

Lee, J.K., Chang, Y., Kwon, H.Y., & Kim, B (2020). Reconciliation of privacy with preventive cybersecurity: the bright internet approach. *Information Systems Frontiers*, 1–13.

Liang, X., Cao, Z., Lin, H., & Shao, J (2009). Attribute based proxy re-encryption with delegating capabilities. In *Proceedings of the 4th international symposium on information, computer, and communications security* (pp. 276–286), https://doi.org/10.1145/1533057.1533094.

Lorünser, T., Slamanig, D., Länger, T., & Pöhls, HC (2016). Prismacloud tools: a cryptographic toolbox for increasing security in cloud services. In *2016 11Th international conference on availability, reliability and security (ARES)* (pp. 733–741). IEEE.

Maesa, D.DF., Mori, P., & Ricci, L (2017). Blockchain based access control. In *IFIP international conference on distributed applications and interoperable systems* (pp. 206–220). Springer.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf.

Ouaddah, A., Abou Elkalam, A., & Ait Ouahman, A (2016). Fairaccess: a new blockchain-based access control framework for the internet of things. *Security and Communication Networks*, *9*(18), 5943–5964.

Pinno, O.J.A., Gregio, A.R.A., & De Bona, L C (2017). Controlchain: Blockchain as a central enabler for access control authorizations in the IoT. In *GLOBECOM 2017–2017 IEEE Global Communications Conference* (pp. 1–6). IEEE.

Priesnitz Filho, W., Ribeiro, C., & Zefferer, T (2019). Privacy-preserving attribute aggregation in eid federations. *Future Generation Computer Systems*, *92*, 1–16.

Provable blockchain (2020). The Provable blockchain oracle for modern DApps. http://provable.xyz/, Accessed 13 January 2021.

RIPEMD160 (2020). https://en.wikipedia.org/wiki/RIPEMD, Accessed 13 January 2021.

Ropsten Testnet Explorer (2020). https://ropsten.etherscan.io, Accessed 13 January 2021.

Sahai, A., & Waters, B. (2005). Fuzzy identity-based encryption. In *Annual international conference on the theory and applications of cryptographic techniques* (pp. 457–473). Springer.

SAML (2020). https://en.wikipedia.org/wiki/SAML_2.0, Accessed 13 January 2021.

Sayeed, S., & Marco-Gisbert, H. (2019). Assessing blockchain consensus and security mechanisms against the 51% attack. *Applied Sciences*, *9*(9), 1788.

SHA-2 (2020). https://en.wikipedia.org/wiki/SHA-2, Accessed 13 January 2021.

Shin, D., & Hwang, Y. (2020). The effects of security and traceability of blockchain on digital affordance. Online Information Review.

Shin, D., & Ibahrine, M. (2020). The socio-technical assemblages of blockchain system: How blockchains are framed and how the framing reflects societal contexts. Digital Policy, Regulation and Governance.

Shin, D.D. (2019). Blockchain: The emerging technology of digital trust. *Telematics and Informatics*, *45*, 101278.

Shin, D.D., Fotiadis, A., & Yu, H (2019). Prospectus and limitations of algorithmic governance: an ecological evaluation of algorithmic trends. Digital Policy, Regulation and Governance.

Shin, D.H. (2010). The effects of trust, security and privacy in social networking: a security-based approach to understand the pattern of adoption. *Interacting with computers*, *22*(5), 428–438.

Solidity (2020). https://solidity.readthedocs.io/en/v0.5.8, Accessed 13 January 2021.

Swan, M. (2015). *Blockchain: Blueprint for a new economy*. Boston: O'Reilly Media Inc.

Wang, G., Liu, Q., & Wu, J (2010). Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In *Proceedings of the 17th ACM conference on Computer and communications security* (pp. 735–737). ACM.

Wang, S., Zhang, Y., & Zhang, Y (2018). A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access*, *6*, 38437–38450.

Web3j SDK (2020). https://web3j.io/, Accessed 13 January 2021.

Antonia Russo is PhD student in computer science at the University Mediterranea of Reggio Calabria. She received her MsC Degree in Telecommunication Engineering from the University Mediterranea of Reggio Calabria in July 2018. Her research interests include security, privacy, and social network analysis.

Gianluca Lax is Associate Professor of Computer Science at the University Mediterranea of Reggio Calabria, Italy. In 2005, he received his PhD in computer science from the University of Calabria. In 2018, he got the habilitation as a Full Professor of Computer Science. His research interests include information security and social network analysis. He is an author of more than 120 papers published in leading international journals and conference proceedings. He serves as a referee for many international journals and is in the program committee of many conferences. He is also included in the editorial board of several international journals and participates in many funded projects.

Baptiste Dromard is a MsC Degree student at Ensicaen, France. His research interest is focused on Blockchain and Bitcoin.

Menad Mezred is a MsC Degree student at Ensicaen, France. His research interest is focused on Blockchain and Ethereum.