

Self-sovereign Management of Privacy Consensus using Blockchain

Francesco Buccafurri^a and Vincenzo De Angelis

DIIES, University Mediterranea of Reggio Calabria, Reggio Calabria, Italy

Keywords: Self-sovereign Identity, Privacy, Consensus, Blockchain.

Abstract: In this paper, we propose a solution implementing a self-sovereign approach to manage privacy consensus in an open domain. The idea is allowing the user to set her policies in a unique way, in such a way that she keeps the full control on her personal data. The goal is achieved by combining blockchain with Attribute-Based Encryption and Proxy Re-encryption. Blockchain is also used to implement the notarization of the critical actions to obtain accountability and non-repudiation.

1 INTRODUCTION


In the recent years, a new paradigm, called Self-Sovereign Identity (SSI), is emerging (Baars, 2016). SSI architectures ensure that users have the full control over their personal data. The idea is that the user is the real owner of personal data, they are stored in a unique place, and the user can establish the right privacy policies just enforcing how the different parties can access them. The benefits of this paradigm mainly regard the effectiveness of the control the user can perform on data. Indeed, one of the major problems of privacy consensus management is the proliferation of sites where data are stored together with the related privacy policies. Typically, for each service provider which needs to use personal data of the customer (which is the most frequent case), a specific data entry and consensus setting is required to the user. This makes infeasible an effective control on data, but also on the compliance of data usage with privacy policies. Also unauthorized transfer of data to third parties is hardly controllable. SSI aims at a sort of *manifest-based* approach, in which the user declares what is possible and what is not, and everyone can verify (even *ex-post*), if data were managed correctly of some abuses occurred.

To reach this goal is not simple. A first problem (say P1) to tackle is that the manifest should not be in clear, because it would be a powerful source of information leakage itself, so an intolerable threat to privacy. A second problem (P2) is that, for the user, enumerating all the policies for every potential service

provider is infeasible, in general. But, on the other hand, we can expect that policies can be expressed as general rules, on the basis of the features of service providers (with possible exceptions). For example, a possible user's rule could be: *all medical data can be accessed by public Health institutions*. A third problem (P3) is that the user should have the possibility to securely publish data and privacy policies in such a way that no trust is required to the party publishing them.

In this position paper (which is related to an industrial research project), we define a solution to the above problem by suitably combining different paradigms. The first is blockchain. Indeed, it is often stated that blockchain can be very powerful to implement SSI-based solutions. However, no effective solution for privacy consensus management (to the best of our knowledge) has been so far proposed. As matter of fact, besides the intuition that blockchain can help SSI, no concrete declination of this statement is available at moment. In our solution blockchain allows us to really give the user the control on personal data and privacy policies in such a way that no party can tamper this information and also critical actions can be notarized to obtain accountability. Basically, blockchain solves problem P3, stated above.

But to deal with the other problems introduced earlier (P1 and P2), we need to integrate blockchain with two advanced cryptographic techniques: Attribute based encryption (ABE) and proxy re-encryption (PRE). Classical Public-key cryptography requires the sender of the message knows exactly who the recipient is. In many situations, it is not important to know the identity of the recipient but it is

^a  <https://orcid.org/0000-0003-0448-8464>

enough that this latter owns some attributes. ABE schemes allow users to decrypt a message, encrypted under a certain policy, if they own the attributes that satisfy such policy. Instead, PRE schemes allow to delegate a third part (proxy) to re-encrypt a ciphertext (i.e. to change the policy associated with it) without exposing the content of plaintext.

Our solution resumes the scheme of public digital identity (Union, 2014), where the role of the Identity Provider is replaced by the Consensus Provider (CP). When a service provider *SP*, during the interaction with a (still anonymous) user, needs her personal data, a procedure is established in which *CP* is required to allow the access to all data compliant with the consensus released by the user for those service providers having the right attributes. Thanks to the blockchain, we can notarize all the critical actions in such a way misbehavior can be detected. As a final remark, we observe that the choice of using an ABAC (Attribute-based-access-control) approach to express privacy consensus is coherent with the application context in which, typically, the user has no reason to allow or deny the access to personal data to a specific service provider. Conversely, the user protects her privacy by stating general rules, mostly related to the scope of the service provider. On the other hand, if the user wants to define a policy specific for a given service provider, it suffices to include, as attribute, just the identity of this service provider.

The structure of this paper is the following. In the next section we contextualize our paper in the related literature. In Section 3, we give some preliminary notions useful for the sequel of the paper. In Section 4, we describe our reference scenario, to better highlight which are the motivations of our proposal. In Section 5, our solution is presented in detail. In Section 6, we sketch some additional features of our model, allowing the notarization of critical actions. Finally, in Section 7, we draw our conclusions.

2 RELATED WORK

In the recent years, a new type of public-key encryption technique, called ABE (Attribute based encryption) has been developed. ABE is characterized by attributes and policies and the decryption of a message is allowed if the policies are satisfied by the attributes. The first ABE scheme was proposed in (Sahai and Waters, 2005), in which the authors present a new type of Identity-based encryption scheme, where the identity is composed of a set of attributes. Subsequently, in (Goyal et al., 2006) and (Bethencourt et al., 2007) respectively, the notions of KP-ABE and

CP-ABE were formalized. The difference is that, in CP-ABE the policy is associated with a ciphertext and the attributes with a secret key belonging to a user, while in KP-ABE the policy is associated with the secret key and the attributes with the ciphertext.

ABE techniques can be integrated with PRE (proxy re-encryption) (Blaze et al., 1998). In PRE a user delegates a semi-trusted proxy to re-encrypt a ciphertext intended for him/her into another ciphertext for a different user. The integration of ABE and PRE is called ABPRE and is useful, for example, in those contexts when a user wants to change the policy associated with the ciphertext (CP-ABPRE) (Liang et al., 2013).

In this paper, we combine ABPREs with blockchain. The notion of blockchain takes origin from the original paper of (Nakamoto et al., 2008). In this survey (Zheng et al., 2018), the reader may find an overview on all the features of blockchain. In (Zyskind et al., 2015; Fan et al., 2018) two solutions are proposed to share private data that use blockchain. However, our proposal is different because we implement access control by leveraging to ABE and PRE schemes. To the best of our knowledge, no proposal combining blockchain and ABPRE can be found in the literature.

3 BACKGROUND

In this section, we provide the technical background necessary to understand our proposal.

Access Structures (Beimel, 1996): Let $\{P_1, P_2, \dots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \dots, P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) \mathbb{A} of non-empty subsets of $\{P_1, P_2, \dots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \dots, P_n\}} \setminus \{\emptyset\}$. The sets in \mathbb{A} are called the authorized sets, whereas the other sets are called the unauthorized sets. \mathbb{A} is also called policy.

CP-ABPRE Scheme: A CP-ABPRE Scheme consists of 6 algorithms:

- **Setup(k):** This algorithm receives a security parameter k and returns a public parameter PK and a master secret key MSK .
- **Encrypt($PK; M; \mathbb{A}$):** This algorithm encrypts a message M under the policy (access structure) \mathbb{A} by using PK . It outputs a ciphertext CT that can be decrypted only by a user who owns the attributes that satisfy \mathbb{A} .

- $\text{KeyGen}(MSK;S)$: This algorithm takes as input a set of attributes S and the master secret key MSK . It outputs a private key SK associated with S .
- $\text{Decrypt}(CT;SK;PK)$: This algorithm takes as input a public parameter PK , a private key SK associated with attributes S and the ciphertext CT encrypted under the access structure \mathbb{A} . If S satisfies \mathbb{A} , the algorithm outputs M .
- $\text{ReKeyGen}(PK,SK;\mathbb{A}')$: This algorithm takes as input a private key SK associated with a set of attributes S and an access structure \mathbb{A}' . It outputs a re-encryption key RK that can be used, by a proxy, to re-encrypt a ciphertext CT , encrypted under a policy \mathbb{A} , into a new ciphertext CT' encrypted under the policy \mathbb{A}' . The re-encryption is allowed only if S satisfies \mathbb{A} .
- $\text{ReEncrypt}(PK,CT,RK)$: This algorithm uses the re-encryption key RK to re-encrypt the ciphertext CT , under a certain policy, into another ciphertext CT' under a new policy.

4 THE REFERENCE SCENARIO

In this section, we describe our reference scenario, to better highlight the motivations of our proposal. The proposal is quite general, because refers to all the cases in which personal data of given user are required by a service provider (i.e., a company, a research institution, etc.), and we want to make the access compliant with privacy consensus stated by the user. Just as example, we consider a medical scenario.

The main entities we consider in our solution are: (1) the user U , (2) the service provider SP , (3) the consensus provider CP . Associated with the user U , a set of personal data exists, which can be composed of different segments. Without loss of generality, we assume that segments are just files and the granularity of privacy rules is at level of files. For example a segment for a user can be represented by her health data. According to a self-sovereign approach, data are stored somewhere, and the user keeps a full control on them. If a service provider want to access some segments of these data, it has to refer to the above repository and must be compliant with privacy consensus stated by the user. The user establish privacy consensus in a sort of manifest, which cannot be public, for privacy reasons, but its management is delegated to a third party, that is the consensus provider CP . It is worth noting that, in our solution, there is no information leakage towards CP . In other words, the role of CP does not give any privilege in terms of right to access personal data of users. CP works in our solu-

tion to implement the way to store somewhere privacy consensus and enforce the application of such rules.

As our solution is based on blockchain and aims to decentralize the most information/management, the repository of personal data is not centralized. Instead, we use the InterPlanetary File System (IPFS) (Benet, 2014). It is a new protocol, based on blockchain, to store and share data in a distributed file system. Blockchain is used as index of such data and to notarize the critical actions.

In our example, a possible consensus rule could be something like: *all health data can be accessed by service provider belonging to the category health institution*.

This means that when a service provider SP belonging to this category is interacting with the user (initially completely anonymous) and needs to access to her health data, an authorization procedure is performed involving the user and CP , to check if the request is compliant with the privacy consensus. The innovation of our solution is that the CP does not perform a classical access control enforcement. When SP requires to access the data, CP does not need to check if it satisfies the policy because it is, intrinsically, performed by the cryptographic ABE scheme used. The only requirement is that SP is able to demonstrate the attributes it holds. The solution includes also a notarization mechanism to prevent misbehavior of all the involved parties. As a final note, we observe that the access rights granted by U are always temporary and she can revoke or change such rights at any time.

5 PROPOSED SOLUTION

In Figure 1, the architecture and the entities involved in our solution are depicted. We introduce the notation used in the following.

1. let U be the user who needs a service s provided by a service provider SP .
2. let SP be the service provider of the service s . To provide s , SP needs the access to some files in F_U .
3. let $F_U = \{f_1, f_2, \dots, f_n\}$ be a list of files owned by U .
4. let $K_U = \{k_1, k_2, \dots, k_n\}$ be a set of symmetric keys (owned by U), where $k_i \in K_U$ is used to encrypt a file $f_i \in F_U$ or to decrypt a file $c_i \in C_{IPFS}$.
5. let $C_{IPFS} = \{c_1, c_2, \dots, c_n\}$ be the list of U 's files encrypted and stored in $IPFS$. Specifically, $C_{IPFS} = \bigcup_{i=1}^n E(k_i, f_i)$.

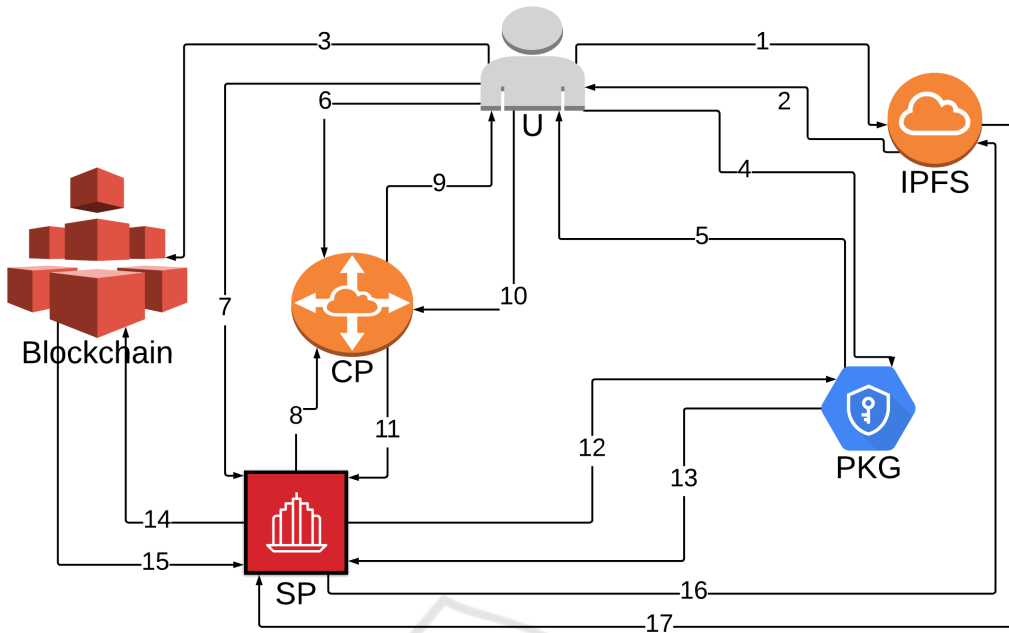


Figure 1: Architecture for the privacy consensus management.

6. let $IPFS$ be the repository containing all the files of the users. We use this acronym to mean that it is implemented by using the InterPlanetary File System (Benet, 2014). When U stores a new encrypted file c on $IPFS$, an index $i_{IPFS}(c)$ is returned that can be used to recover the file c .
7. let Add_U be the blockchain address of U .
8. we model a transaction T as a tuple $\langle id_T, Add_{src}, Add_{dest}, data \rangle$, where id_T is the identifier (usually, it is the digest of the transaction), src and $dest$ are the sender and receiver of the transaction, respectively, and $data$ is the payload.
9. let PKG be the entity responsible for the generation of ABE public key PK and ABE master secret key MSK . Moreover, it generates the ABE private keys SK for U and SP .
10. $E(k, f)$ denotes the encryption of a file f with the symmetric key k .
11. $D(k, c)$ denotes the decryption of a ciphertext c with the symmetric key k .
12. let CP be the consensus provider. It manages the access policies and plays the role of proxy re-encryptor.
13. $Setup(s), Encrypt(PK, M, \mathbb{A}), Decrypt(CT, SK), KeyGen(MSK, S), ReKeyGen(PK, SK, \mathbb{A}'), ReEncrypt(PK, CT, RK)$ are the ABPRE algorithms as defined in Sec.3.
14. let V_{CP} be a set, owned by CP , that contains the symmetric keys in K_U encrypted, with ABE, under a certain policy. Specifically, $V_{CP} = \bigcup_{i=1}^n Encrypt(PK, k_i || id_T, \mathcal{A}_I)$ where \mathcal{A}_I is the policy associated with the file f_i and id_T is the identifier of the blockchain transaction generated when the file f_i (encrypted) has been stored on $IPFS$.
15. let RK_{CP} be a set of re-encryption keys owned by CP . Specifically, $RK_{CP} = \bigcup_{i=1}^n ReKeyGen(PK, SK_U, \mathcal{A}'_I)$, where SK_U is the ABE secret key associated with U 's attributes and \mathcal{A}'_I is a new policy associated with the file f_i . So, each $rk_i \in RK_{CP}$ is computed by U and sent to CP .

To be more accurate, we should include in the architecture the attribute provider AP that are in charge on checking if SP or U own the declared attributes. For simplicity, we assume that this function is performed, directly, by the PKG . In this position paper, this aspect is not treated in detail.

Now, we describe the steps carried out by the different actors of our scenario. These steps are also summarized in Figure 1.

Step 1: Setup. PKG invokes $Setup(s)$ (where s is an implicit security parameter) and generates PK and MSK . U and CP shall exchange a seed to initialize a PRNG. This PRNG is used to generate a not guessable number that identifies a request of U . In this way, the different requests of U are unlinkeable.

Step 2: File storage. U wants to share a new file f . First, U updates $F_U \leftarrow F_U \cup \{f\}$, selects a new symmetric key k , updates $K_U \leftarrow K_U \cup \{k\}$ and computes $c = E(k, f)$. Now, U stores c on $IPFS$ and obtains $i_{IPFS}(c)$ (messages 1-2). $IPFS$ updates $C_{IPFS} \leftarrow C_{IPFS} \cup \{c\}$. Finally, U generates a self-transaction (message 3) $T = \langle id_T, Add_U, Add_U, data \rangle$ where $data = (i_{IPFS}(c), HMAC(k, f))$.

Step 3: Delegation to CP. This step is performed right after the Step 2. U contacts the PKG and declares the set of attributes P that he/she owns (message 4). At least, P is composed of the attribute ID_U that identifies unequally U . PKG checks that U really owns all the attributes in P , and if the check is positive, it returns the ABE private key SK_U associated with P (message 5). Observe that this request to PKG is performed by U just once. Indeed, SK_U can be used for future files. At this point, U chooses a policy \mathbb{U} that requires the possession of the attribute ID_U and computes $v = Encrypt(PK, k || id_T, \mathbb{U})$. So, in word, v is the symmetric key, used to encrypt the file f , encrypted with ABE in a way that only U can decrypt it (id_T is used in the next Step).

Now, U chooses a policy \mathcal{A}' , and invokes $rk = ReKeyGen(PK, SK_U, \mathcal{A}')$. So, U sends the tuple (v, rk) to CP (message 6) that updates $V_{CP} \leftarrow V_{CP} \cup \{v\}$ and $RK_{CP} \leftarrow RK_{CP} \cup \{rk\}$

Step 4: Service request. U requires a service s provided by SP . He/She generates a random number R (through the PRNG) that identifies this request and sends (ID_{CP}, R) to SP (message 7). ID_{CP} is the identifier of the CP in charge to manage U 's policies. Through ID_{CP} , SP contacts CP and sends (ID_{SP}, R) (message 8), where ID_{SP} is the identifier of SP . Now, CP verifies that U really has asked SP for the service s . So, CP extracts R' from the PRNG and checks that $R = R'$, then it performs a mutual authentication with U (by means of a challenge-response mechanism)(messages 9-10). At this point, CP computes $Q_{SP} = \bigcup_{i=1}^n ReEncrypt(PK, v_i, rk_i)$. In word, Q_{SP} contains all the symmetric keys used by U , re-encrypted each one with an appropriate policy. So, CP sends Q_{SP} to SP (message 11). Now, in order to decrypt the keys in Q_{SP} , SP needs the ABE private key SK_{SP} associated with its attributes. The procedure is similar to Step 3. So, SP contacts PKG and declares a set of attributes P' (message 12), PKG checks if SP owns these attributes and send SK_{SP} (message 13). At this point, for each $q_i \in Q_{SP}$, SP invokes $Decrypt(q_i, SK_{SP})$ and retrieves $k_i || id_{T_i}$. Obvi-

ously, Q_{SP} contains all the private keys of U , but SP can access (i.e. decrypt) to only some of them (those for which it satisfies the policy). Now, for each $q_i \in Q_{SP}$ that SP is able to decrypt, it looks for the transaction with identifier id_{T_i} and retrieves $(i_{IPFS}(c_i), HMAC(k_i, f_i))$ (messages 14-15). Finally, SP contacts $IPFS$ and asks for the file c_i at the position $i_{IPFS}(c_i)$ (message 16-17). Once SP obtain c_i , it retrieves the file $f_i = D(k_i, c_i)$ and checks the integrity computing $HMAC(k_i, f_i)$.

Step 5: Policy change and Revocation. Suppose U wants to change the policy \mathcal{A}_I associated with a file $f_i \in F_U$, with a new policy \mathcal{A}'_I . We can distinguish two cases:

- All users that satisfy \mathcal{A}'_I , also satisfy \mathcal{A} . In this case, the new policy allows new users to access the file and maintains the access right of old users. So, it is sufficient that U invokes $rk'_i = ReKeyGen(PK, SK_U, \mathcal{A}'_I)$ and sends rk'_i to CP . CP updates $rk_i \leftarrow rk'_i$.
- The policy \mathcal{A}'_I is generic. In this case, if U doesn't want to give (wants revoke) the access right to old users, he has to change the symmetric key k_i . So, U remove the c_i from $IPFS$ and starts Step2 and Step3 again.

6 NOTARIZATION

In this section, we describe an additional feature we can include in our solution. It is the notarization of a number of the critical operations aimed to obtain accountability and non-repudiation. Obviously, notarization is obtained by leveraging the properties of blockchain. First, we assume that the user U and the Consensus Provider CP , in a preliminary stage, exchange their blockchain addresses through a non-repudiable way. We want to notarize two critical actions: (1) the policy setting/change done by the user, and (2) the policy enforcement done by CP .

The benefit of the notarization (1) is that the misbehavior of both U and CP can be detected. For example, it could happen that when U changes the policies, CP could keep the old policies. Notarization is obtained (concerning the policy of a file f), through the publication on the blockchain of a transaction $T = \langle id_T, Add_U, Add_{CP}, data \rangle$, where $data = (H(rk || f))$. H is a hash function and rk is the new re-encryption key generated by U (see Step 5 in section 5). Therefore, in the previous example, if CP doesn't use the new policy (i.e. the new rk), U can prove it through rk and f .

Notarization (2) prevents the misbehavior of CP

or SP . In this case, we want to be sure that, when CP sends Q_{SP} (message 11) to SP , it includes all and only the symmetric keys of U . This goal is reached by generating a transaction $T = (id_T, Add_{CP}, Add_{SP}, data)$ where $data = (H(q_1 || q_2 || \dots || q_n))$ and $\forall i \in [1, n], q_i \in Q_{SP}$.

We highlight that other action can be notarized, as those regarding the interaction between SP and AP . In this position paper, we do not care this aspect which will be treated in the next steps of our research.

7 CONCLUSIONS

In this paper, a self-sovereign-based approach to manage privacy consensus to access personal data is provided. The solution leverages the combination of blockchain with some advanced cryptographic schemes, like attribute-based encryption and proxy re-encryption. As message exchange flow, our protocol resumes the scheme of federated authentication, like SAML2 (Lockhart and Campbell, 2008) or OpenId Connect (Sakimura et al., 2014). So, a full implementation of our protocol can be done by extending the features of one of the protocols mentioned earlier. In a real-life adoption of our solution, we should also understand who play the role of the various entities, in particular CP , PKG and AP . While no high-level trust is required to CP , which can be a company (as the identity provider in public digital identity systems), AP and PKG play a more critical role. Therefore, they should be government institutions, or solutions to decentralize also these function should be studied. Observe that this problem exists also for the attribute providers of public digital identity systems compliant with the EU regulation (Union, 2014). As a future work, we plan to address the above problems (i.e., implementation and real-life setting), together with a careful security analysis to state more formally which are the security features of our solution,

ACKNOWLEDGEMENTS

This paper is partially supported by the project “SecureOpenNets-Distributed Ledgers for Secure Open Communities”, funded by Ministry of Research and Education (MIUR), project id ARS01_00587.

REFERENCES

Baars, D. (2016). Towards self-sovereign identity using blockchain technology. Master’s thesis, University of

- Twente.
- Beimel, A. (1996). *Secure schemes for secret sharing and key distribution*.
- Benet, J. (2014). Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*.
- Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *2007 IEEE Symposium on Security and Privacy (SP '07)*, pages 321–334.
- Blaze, M., Bleumer, G., and Strauss, M. (1998). Divertible protocols and atomic proxy cryptography. In Nyberg, K., editor, *Advances in Cryptology — EUROCRYPT'98*, pages 127–144, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Fan, K., Wang, S., Ren, Y., Li, H., and Yang, Y. (2018). Medblock: Efficient and secure medical data sharing via blockchain. *Journal of Medical Systems*, 42(8):136.
- Goyal, V., Pandey, O., Sahai, A., and 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, CCS '06*, pages 89–98, New York, NY, USA. ACM.
- Liang, K., Fang, L., Susilo, W., and Wong, D. S. (2013). A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security. In *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, pages 552–559. IEEE.
- Lockhart, H. and Campbell, B. (2008). Security assertion markup language (saml) v2.0 technical overview. *OASIS Committee Draft*, 2:94–106.
- Nakamoto, S. et al. (2008). Bitcoin: A peer-to-peer electronic cash system.
- Sahai, A. and Waters, B. (2005). Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 457–473. Springer.
- Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and Mortimore, C. (2014). Openid connect core 1.0 incorporating errata set 1. *The OpenID Foundation, specification*.
- Union, E. (23 July 2014). Regulation EU No 910/2014 of the European Parliament and of the Council. <http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX\%3A32014R-0910&from=EN>.
- Zheng, Z., Xie, S., Dai, H.-N., Chen, X., and Wang, H. (2018). Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375.
- Zyskind, G., Nathan, O., and Pentland, A. . (2015). Decentralizing privacy: Using blockchain to protect personal data. In *2015 IEEE Security and Privacy Workshops*, pages 180–184.