Transparent Personal Data Processing: The Road Ahead

Piero Bonatti[1], Sabrina Kirrane[2], Axel Polleres[2,3], and Rigo Wenning[4]

[1] Universita' di Napoli Federico II, Italy
[2] Vienna University of Economics and Business, Vienna, Austria
[3] Complexity Science Hub Vienna, Vienna, Austria
[4] W3C, Sophia-Antipolis, France

Abstract. The European General Data Protection Regulation defines a set of obligations for personal data controllers and processors. Primary obligations include: obtaining explicit consent from the data subject for the processing of personal data, providing full transparency with respect to the processing, and enabling data rectification and erasure (albeit only in certain circumstances). At the core of any transparency architecture is the logging of events in relation to the processing and sharing of personal data. The logs should enable verification that data processors abide by the access and usage control policies that have been associated with the data based on the data subject's consent and the applicable regulations. In this position paper, we: (i) identify the requirements that need to be satisfied by such a transparency architecture, (ii) examine the suitability of existing logging mechanisms in light of said requirements, and (iii) present a number of open challenges and opportunities.

1 Introduction

The European General Data Protection Regulation (GDPR), and *Articles 12-15* in particular, calls for technical means to support the obtaining of explicit consent from data subjects and the provision of transparency with respect to personal data processing and sharing. In order to provide said transparency, companies need to record details of personal data processing activities and personal data transactions (i.e. who shared what data with whom, for what purpose and under what usage conditions). From a technical perspective there is a need for a transparency architecture that records metadata (i.e. policies, event data, context), that can be used to verify that data is processed according to the wishes of the data subject and the applicable regulations.

From a high level perspective such a transparency architecture needs to enable: (i) data subjects to verify that data processors are complying with usage policies; and (ii) data processors to demonstrate that their business processes comply both with the policies accepted by the data subject and the obligations set forth in the GDPR. As a first step towards realising this goal, in this position paper, we examine the suitability of existing logging and transparency mechanisms as the basis for developing such a system. There exists a variety of logging mechanisms that either represent events in local logs [2, 7, 10, 14, 15], in global

logs entrusted to one or more third parties [1, 6, 11, 12], or distribute event logging across a number of peers [16, 19, 21]. Ideally it should be possible to use some of these logging mechanisms together with access/usage policies in order to automatically verify compliance of existing business processes with the GDPR. The contributions of the paper can be summarised as follows: we (i) identify a list of requirements relevant for transparent processing and sharing of personal data; (ii) examine the degree of support, with respect to said requirements, offered by the different logging architectures (i.e. ledgers); and (iii) discuss the open research challenges and opportunities.

The remainder of the paper is structured as follows: *Section* 2 discusses the requirements that are relevant for data usage transparency frameworks. *Section* 3 identifies a number of candidate approaches and examines their suitability in terms of support for the identified requirements. *Section* 4 points to several Resource Description Framework (RDF) vocabularies that can be used to represent data processing and sharing events and highlights a number of open research questions. Finally, we present our conclusions and some interesting directions for future work in *Section* 5.

2 Data Processing Transparency Requirements

Before discussing the different logging architectures (i.e. ledgers) we first provide a concrete motivating scenario and identify several data processing and sharing transparency and robustness requirements.

2.1 A Motivating Scenario

Sue buys a wearable appliance for fitness tracking from BeFit. She is presented with an informed consent request, comprised of a data usage policy that describes which data shall be collected, and how they will be processed and transmitted in order to give her fitness-related information. The policy says that the device records biomedical parameters such as heart rate; these data are stored in BeFit's cloud; and processed for two purposes: (i) giving Sue feedback on her activity, such as calories consumption; (ii) (optional) creating an activity profile that will be shared with other companies for targeted ads related to fitness. Sue opts in for (ii) in order to get a discount. The usage policy, signed by both Sue and BeFit, is stored in a *transparency ledger*. After one year, the device stops working. After two years, Sue starts receiving annoying SMS messages from a local gym that advertise its activities. Fortunately, all the data collection, processing, and transmission operations have been recorded in the transparency ledger. By querying the ledger, Sue discovers the following facts: (i) the gym has an activity profile referring to Sue, that due to the appliance's malfunctioning reports that she is not doing any physical exercise; (ii) the gym received the profile from BeFit, associated with a policy that allows the gym to send targeted ads to Sue based on the profile; (iii) BeFit built the profile by mining the data collected by the appliance; and (iv) all these operations are permitted by the

consent agreement previously signed by Sue and BeFit. Using the information contained in the ledger, BeFit and the gym can prove that they used Sue's data according to the agreed purposes. However, Sue can now ask both BeFit and the gym to delete all of her data. The information contained in the ledger indicates precisely which pieces of information she is referring to, so they can be automatically deleted in real time.

2.2 Ledger Functionality and Robustness

In order to enable scenarios such as that described above and to provide the technical basis for companies to demonstrate that their business processes comply with the consent, transparency, rectification and erasure obligations specified in the GDPR, a fundamental first step is to create a ledger of all data transactions (i.e. who shared what data with whom, for what purpose and under what usage conditions) and to record what happened to the data (e.g. processing, anonymisation, aggregation). In order to provide transparency with respect to data processing to the data subject, while at the same time allowing companies to demonstrate that they are complying with the regulation the following core functions are required.

Ledger functionality

Completeness: All data processing and sharing events should be recorded in the ledger.

Confidentiality: Both data subjects and companies should only be able to see the transactions that involve their own data.

Correctness: The records stored in the ledger should accurately reflect the processing event.

Immutability: The log should be immutable such that it is not possible to go back and reinvent history.

Integrity: The log should be protected from accidental and/or malicious modification.

Interoperability: The infrastructure should be able to transcend company boundaries, in the sense that the data subject should be able to easily combine logs that they get from multiple companies.

Non-repudiation: When it comes to both data processing and sharing events it should not be possible to later deny that the event took place.

Rectification & Erasure: It should be possible to rectify errors in the stored personal data and/or delete data at the request of the data subject.

Traceability: In the case of processing it should be possible to know about any previous processing of the data. As such it should be possible to link events in a manner that supports traceability of processing.

Ledger Robustness

Availability: Availability is the process of ensuring the optimal accessibility and usability of the ledger irrespective of whether the log is stored locally or globally. Here there is also a link to security as it is imperative that a breach of security does not hinder ledger operations.

Table 1. Candidate architectures and ledger functionality gap analysis

	Local Log	Global Log + TTP	Global Log + P2P
Completeness	-	-	-
Confidentiality	MAC [2, 7, 14, 15], FssAgg [10], PKI [7, 10]	MAC [1, 6, 11, 12], PKI [20], unlinkability [6, 11, 12]	MAC [15], PKI[16], compound identities [15, 21]
Correctness	-	-	-
Immutability	cipher chains [2], hash chains [7, 15]	hash chains [7, 15]	network of peers [16, 19] blockchain [21]
Integrity	forward integrity [2, 7, 10, 14, 15] MAC security proof [2]	forward integrity [1, 6, 11, 12]	forward integrity [15]
Interoperability	-	-	-
Non-repudiation	-	-	-
Rectification & Erasure	-	-	-
Traceability	-	event trails [20]	-

Table 2. Candidate architectures and ledger robustness gap analysis

	Local Log	Global Log + TTP	Global Log + P2P
Availability	-	-	-
Performance	logging & verification [2, 7], signature generation & verification [10]	logging [11, 12], throughput [11, 12]	-
Scalability	encrypting records [7, 10]	-	-
Storage	key & signature [10]	resource restricted devices [1]	-

Performance: When it comes to the processing of the event data, various optimisations such as parallel processing and/or indexing should be used to improve processing efficiency.

Scalability: Given the volume of events and policies that will need to be handled, the scalability of event data processing is a major consideration.

Storage: In order to reduce the amount of information stored in the log, the data itself should be stored elsewhere and only a hash of the data and a pointer to the actual data itself should be stored in the ledger.

3 Candidate Transparency Ledgers

The overarching goal of this section is to examine the potential solutions proposed in the literature in order to understand the strengths and limitations of existing proposals and to identify challenges that still need to be addressed. A summary of the degree of support offered by the candidate transparency architectures is provided in *Table* 1 and *Table* 2.

3.1 The Status Quo

When it comes to the persistence of provenance records there are three high level options, that are not necessarily mutually exclusive: each company maintains a local ledger, which may be backed up remotely; a global ledger could

be maintained by one or more trusted third parties; or a global ledger could be distributed across a number of peers.

Local ledger Each peer could store its provenance records locally, including information pertaining to data sharing (both incoming and outgoing). While, remote logging to a trusted third party (TTP) could be used to guarantee recoverability of data if the machine where the log is stored is compromised. Bellare and Yee [2] and Schneier and Kelsey [15] demonstrated how a secret key signing scheme based on Message Authentication Codes (MACs) together with a hashing algorithm can be used to generate chains of log records that are in turn used to ensure log confidentiality and integrity. MACs are themselves symmetric keys that are generated and verified using collision-resistant secure cryptographic hash functions. Bellare and Yee [2] discuss how a MAC secret key signing scheme together with evolving MAC keys (whereby each record is encrypted with a different key that is derived from the old key) can be used to ensure: (i) the confidentiality of the log; (ii) that previous log entries cannot be changed; and that (iii) the deletion of a log entry can be detected. In such a scenario the base MAC key, which is needed to verify the integrity of the log is entrusted to a TTP. Schneier and Kelsey [15] also use MACs however the log is composed of hash chains as oppose to cipher block chains. Whereas, Holt [7] propose an alternative that combines public key cryptography with hash chains. These approaches are further enhanced by Ma and Tsudik [10] who demonstrate how individual log entry signatures can be combined into a single aggregate signature that can be used to verify the component signatures and to protect against log truncation. While the previously mentioned works focused on logging in general, Sackmann et al. [14] apply it specifically to data protection by demonstrating how a secure logging system can be used for privacy-aware logging. Additionally, they introduce the "privacy evidence" concept and discuss how such a log could be used to compare data processing to the users privacy policy.

When it comes to the robustness requirements, both Bellare and Yee [2] and Holt [7] evaluate the performance and scalability of the proposed logging and verification algorithms, while Ma and Tsudik [10] compare alternative signature generation and verification algorithms.

Global Ledger and Trusted Third Party: Alternatively, the ledger may contain provenance records that are maintained by one or more TTPs. Accorsi [1] demonstrate how MAC-based secure logging mechanisms can be tailored so that they can be used by resource restricted devices that may need to log data remotely. Wouters et al. [20] highlight the fact that data often flows between different processes, and as such events cannot be considered in isolation, thus giving rise to the need to store a trail of events. The authors demonstrate how public key cryptography can be used to log events in a manner whereby the data subject can verify the process status. Hedbom et al. [6], Peeters et al. [11], Pulls et al. [12] also provide logging mechanisms that provides transparency to data subjects. The protocol, which is based on MAC secure logging techniques,

ensures confidentiality and unlinkability of events and is designed so that it can be distributed across several servers. In the case of [11, 12], each log is composed of a user block, a processor block and the encrypted data. A trusted third party is responsible for generating the MAC, encrypting it with the users public key, signing it with their own private key and sending it to the data subject via the data processor. The data processor block is generated in a similar manner. Both the log and the personal data are encrypted in a manner that only the data subject and the processor can access them. In the case of data sharing a new blinded public key is created (in a manner such that the data subjects private key can decrypt any data encrypted with the blinded public key). The blinded key, which will be used by the second data processor, also serves to ensure the unlinkability of the logs.

Peeters et al. [11], Pulls et al. [12] both evaluate the performance of the proposed algorithms and examine the logging throughput from a local and a remote perspective. The authors conclude that encryption and signing are expensive operations and as such the log entry generation time does not scale linearly with the size of the logged data. They also highlight that the decryption and verification processed are also expensive.

Global Ledger and Peer-to-Peer network: Alternatively, the ledger may be distributed across several physical ledgers (i.e. a virtual global ledger), whereby provenance records are replicated by each peer. Schneier and Kelsey [15] highlight the vulnerability associated with using a single TTP and discuss how n untrusted machines could be used to replace the TTP, with m untrusted machines required to reproduce the base MAC secret key. Weitzner et al. [19] also discuss how transparency and accountability can be achieved via distributed accountability peers that communicate using existing web protocols. These accountability peers would be responsible for mediating access to data, maintaining audit logs and facilitating accountability reasoning. Unfortunately the authors only touch upon the required features and no concrete architecture is proposed. Seneviratne and Kagal [16] build on this idea by describing how a distributed network of peers can be used to store a permanent log of encrypted transactions. The replication of log entries at each peer optimises both redundancy and availability. Although the authors describe how a distributed network of peers can be used to store a permanent log of transactions, they focus primarily on helping users to conform to policies by highlighting not only usage restrictions but also the implications of their actions, as opposed to investigating the functional and technical challenges of the proposed transparency architecture itself. An alternative distributed architecture based on blockchain technology, which can be used to manage access to personal data is proposed by Zyskind et al. [21]. The authors discuss how the blockchain data model and Application Programming Interfaces (APIs) can be extended to keep track of both data and access transactions. Data that is encrypted using a shared encryption key, is sent to the blockchain, which subsequently stores the data in an off-blockchain key value store and a pointer to the data in the form of a hash in the public ledger. Compound identities are used to ensure that only the user and service providers that have been granted

access to the data can decrypt the data. One of the primary drawbacks is the fact that the authors focus on how to repurpose the blockchain as an access-control moderator as opposed to exploring the suitability of the proposed architecture for data transparency and governance.

In comparison to local or global approaches that employ a third party the robustness of the proposed approaches has not been explored to date, therefore it is difficult to assess the effectiveness of P2P ledgers or blockchains from a non-functional perspective.

3.2 Gap Analysis

Although the main goal of the analysis was to investigate the opportunities and the limitations of each of the candidate architectures, in the end we were able to observe that the primary technical challenges are common across all candidate architectures (cf. *Table* 1 & *Table* 2).

Correctness, Completeness & Non-Repudiation: Although both *correctness* and *completeness* are very desirable features, irrespective of the choice of architecture, when it comes to data processing events neither can be guaranteed as there is no way to prevent companies from logging incorrect information or not entering the information into the log. Although fair exchange protocols could potentially be used to ensure *non-repudiation* of data transactions (i.e. neither party can deny the transaction took place), to date they have not been used in connection with existing logging mechanisms.

Confidentiality & Integrity: The combination of MAC together with cipher or hash chains appears to be the prevailing mechanism used to ensure the confidentiality and forward integrity of logs. Although Schneier and Kelsey [15] highlight that it could be feasible to replace the TTP with n untrusted machines whereby any m are required to reproduce the base MAC secret key, no concrete details are provided. Additionally, in the context of our use case the secure logging verification schemes would need to be extended to cater for *rectification & erasure* without affecting the overall integrity of the log.

Immutability, Rectification & Erasure: Although it should not be possible for a company to go back and reinvent history, the GDPR stipulates that data subjects have the right to *rectification & erasure* (often referred to as the right to be forgotten). This could potentially be seen as a hard delete whereby the data needs to be erased from both the system and the logs. This would mean that we need to be able to update and delete records from the log without affecting the overall integrity of the log. One potential solution would be to employ a cryptographic delete and to provide support for updates via versioning.

Interoperability & Traceability: Another consideration is the interoperability of the log with other logs. Considering that existing logging research has primarily focused recording operating system and application events it is not surprising that interoperability has received very little attention to date. Although there has been some research on *traceability*, the focus has primarily been on linking processing events in a single log.

Performance & Scalability: Considering the potential volume of events that will need to be handled by the transparency ledger, the scalability of existing logging mechanisms will be crucial to their adoption. When it comes to the processing of event data, various optimisations such as parallel processing and/or indexing may improve processing efficiency. Data transfer speed could be improved via exchanging a compressed version of the data payload. Inherently querying and updating logs over distributed databases is a computational challenge.

Storage: In practice it may not be feasible for a single log server or each peer in a distributed network to store all provenance records. One possibility is to split the provenance records into multiple ledgers, distributed among TTPs or peers. However, such an architecture would need to be fault tolerant in the case of peers disconnecting from the network. Relevance criteria and careful forgetting may help too, insofar as storage requirements may be reduced by storing only the information that is needed for compliance checking in the specific domain of interest, and deleting other information.

Availability: Clearly from an availability perspective it is important that the best practices are employed in order to protect the security of the log host. Additionally the log should be backed up to a secure location on a regular basis. It is worth noting that when it comes to log recovery, rather than relying on a TTP a hash of the log could be submitted to a publicly available blockchain (such as Bitcoin). However, unlike trusted third parties public blockchains do not come with Service Level Agreements (SLAs).

4 Challenges and Opportunities

Although in this paper we primarily focus on transparency, our long term goal is to use the ledger together with access/usage policies in order to automatically verify compliance of existing business processes with the GDPR, to this end it is necessary to model both policies and events in a machine readable manner.

4.1 The Ledger

The Resource Description Framework (RDF), which underpins the Linked Data Web (LDW), is used to represent and link information, in a manner which can be interpreted by both humans and machines. Particularly, the power of RDF is revealed in combination with agreed and extensible meta-data vocabularies to describe provenance and events related to data records in a log as metadata, in semantically unambiguous terms. By employing RDF techniques to represent the provenance events stored in the ledger we shall be able to support not only interoperabilily between ledgers, but also traceabiliy between events in a manner that facilitates automatic compliance checking. To this end, there are a number of existing vocabularies that can be adapted/ extended. For example the *PROV*[5] and *OWL-Time*[6] ontologies can be used to represent *provenance* and

[5] PROV,https://www.w3.org/TR/prov-overview/
[6] OWL-Time,https://www.w3.org/TR/owl-time/

temporal information respectively. The former may require extensions of PROV to model particular aspects related to processing of personal data. The latter is particularly relevant if ledger-information is distributed. For example, when tracking audit trails potentially distributed over different systems, synchronisation of timestamps and ensuring sequentiality are major issues. Apart from the actual representation of time, reasoning and querying about time and temporal aspects is still an issue that needs more research in the Semantic Web arena. Different proposals for temporal extensions of RDF and querying archived, temporal information in RDF exist, cf. for instance [5] and references therein. Additionally there exists a number of general event vocabularies such as the *Event*[7] ontology and the *LODE*[8] ontology [13] that could potentially be adapted/extended in order to model our data processing *events*.

An additional benefit of Linked Data is that it provides a simple, direct way of associating policies with data. However, such integration needs to be done in a way that ensures scalability. Several techniques can be exploited for this purpose. As an example, we mention knowledge compilation approaches that "compile" semantic metadata into a compact but self-contained policy that can be more efficiently enforced, without any further access to the knowledge repository (cf. the approaches based on partial evaluation in [3]). The usage of RDF and URIs shall enable the deployment of a linked network of distributed ledgers instead of a single, monolithic (central or P2P ledger). Here it would be interesting to look into efforts for modularising and linking between distributed ledgers such as the recent interledger protocol [8] proposal.

4.2 Ledger Integrity and Reliability

Ensuring the ledger's integrity and reliability is of course essential for compliance checking and for enhancing the subjects' trust in the transparency architecture. Reliability is partly the result of *voluntary compliance*. In the countries with strong data protection regulations, due to the sanctions and the loss of reputation and customers that may result from data abuse, data processors are willing to comply with the regulations, and feel the need for technical means to ensure compliance. In such scenarios, a correct and complete ledger is an extremely useful tool for the data processors, that can exploit it both for verifying their internal procedures, and for demonstrating compliance to data subjects and data protection authorities. This incentivises the creation and maintenance of a correct and complete ledger. As a further incentive to correctness, the event records should be signed by the parties involved in the recorded operation. In this way, the ledger's records become formal declarations that constitute evidence with legal strength (in the countries where digital signatures have legal value), that may be exploited in case of disputes. As a special case, some of the ledger's records may represent data usage consent declaration, in the form of a usage policy signed by the data subject and the data processor. Such records are

[7] Events,http://motools.sourceforge.net/event/event.html
[8] LODE, http://linkedevents.org/ontology/

very close to a contract that none of the two parties can repudiate, due to the properties of digital signatures.

Creating a reliable record for joint operations, and creating records with multiple "simultaneous" signatures, require the adoption of *fair exchange protocols* to guarantee that the operation is completed (e.g. data are transferred) if and only if all the involved parties sign the record and the record is included in the ledger. An extensive survey of fair exchange protocols can be found in [9]. Ideally, the protocol should not involve centralised nodes such as TTP, but the existing approaches of this kind, based on multiparty computations, currently do not scale to the volume of data expected in the scenarios of interest. There are, however, protocols with *offline TTP*, that involve the trusted third party only in case of malfunctioning (like lost or corrupted messages) or protocol violations. As of today, we regard such protocols as the most promising.

4.3 Immutability, Rectification & Erasure

When it comes to transparent personal data processing *immutability* is a very desirable feature as it can be used by companies to prove that they have not gone back and reinvented history. However, said immutability seems to be in direct contention with the right to *rectification and erasure* according to the GDPR. Considering the focus of this paper, we restrict our discussion to the rectification and erasure of the log entries and do not give any special consideration to the Line of Business (LOB) application. By only storing a hash of the data and a pointer to the actual data itself in the ledger it is possible to decouple the data from the log and indeed delete data. Another motivation for doing so is the *storage* requirements can be reduced considerably. In the case of rectification it may suffice to update data in the LOB application(s) and enter a new record in the log indicating that the data was updated at the request of the data subject, including a reference to the old – deleted – records hash that confirms that said record was updated in mutual agreement. Likewise, in terms of erasure, we assume that there are scenarios like rectification where it will suffice to delete data from the LOB application(s) and enter a new record in the log indicating that the data was deleted at the request of the data subject. Although this would result in a dangling pointer from the initial log entry by following the audit trail it would be possible to find out that the dangling pointer is the result of an authorised delete. However, there may also be scenarios where delete means a hard delete that needs to be propagated to the log (e.g., where it is possible to identify the individual from the log entry). One option would be to investigate the application of cryptographic deletes (where the old data should not be available anymore) to the ledger. However, it would need to be possible to distinguish between authorised deletes (at the request of the data subject) and log tampering. As such, any delete or update request needs to be strongly coupled with a request from the data subject. So far, cryptographic deletion has been considered only in cloud computing environments, where files are replicated across virtual and physical nodes, and whatever remains of the files after their standard deletion (which is logical) could be later recovered by an attacker, cf.

[4, 17, 18]. We propose a novel use of cryptographic deletion as a means to harmonise mandatory preservation requirements and the right to deletion, so as to avoid extreme solutions where one requirement overrides the other.

5 Conclusion and Future Work

Transparency with respect to the collection, processing and sharing of personal data is a key enabler for data controllers and processors to achieve GDPR compliance. In this paper, we identified several requirements that are important for enabling transparent processing of personal data at scale. Following on from this, we analysed a number of candidate logging mechanisms and discussed their suitability in light of said requirements. Based on the gaps highlighted by this analysis, we discussed some of the open challenges and opportunities for future research. In particular we identified at least three interesting questions that call for more work: how to ensure ledger interoperability and usage traceability across organisation borders; how to obtain ledger integrity and reliability; and how to reconcile the conflict between the log immutability requirement and the data subjects' right to rectification and erasure. In future work, we will develop a system that enables data subjects to associate sticky usage policies with their personal data and companies to demonstrate compliance both with the usage policies specified by the data subject and obligations set forth in the GDPR.

Acknowledgments. Supported by the European Union's Horizon 2020 research and innovation programme under grant 731601.

References

1. R. Accorsi. On the relationship of privacy and secure remote logging in dynamic systems. In *IFIP International Information Security Conference*, 2006.
2. M. Bellare and B. Yee. Forward integrity for secure audit logs. Technical report, Technical report, Computer Science and Engineering Department, University of California at San Diego, 1997.
3. P. Bonatti, S. De Capitani di Vimercati, and P. Samarati. An algebra for composing access control policies. *ACM Transactions on Information and System Security (TISSEC)*, 5(1), 2002.
4. C. Cachin, K. Haralambiev, H. Hsiao, and A. Sorniotti. Policy-based secure deletion. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13*, 2013.
5. J. D. Fernández Garcia, J. Umbrich, M. Knuth, and A. Polleres. Evaluating query and storage strategies for RDF archives. In *12th International Conference on Semantic Systems (SEMANTICS)*, ACM International Conference Proceedings Series, 2016.

6. H. Hedbom, T. Pulls, P. Hjärtquist, and A. Lavén. Adding secure transparency logging to the prime core. In *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life*, 2009.
7. J. E. Holt. Logcrypt: forward security and public verification for secure audit logs. In *Proceedings of the 2006 Australasian workshops on Grid computing and e-research-Volume 54*, 2006.
8. A. Hope-Bailie and S. Thomas. Interledger: Creating a standard for payments. In *Proceedings of the 25th International Conference Companion on World Wide Web*, 2016.
9. S. Kremer, O. Markowitch, and J. Zhou. An intensive survey of fair nonrepudiation protocols. *Computer communications*, 25(17), 2002.
10. D. Ma and G. Tsudik. A new approach to secure logging. *ACM Transactions on Storage (TOS)*, 5(1), 2009.
11. R. Peeters, T. Pulls, and K. Wouters. Enhancing transparency with distributed privacy-preserving logging. In *ISSE 2013 Securing Electronic Business Processes*. Springer, 2013.
12. T. Pulls, R. Peeters, and K. Wouters. Distributed privacy-preserving transparency logging. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, 2013.
13. M. Rinne, E. Blomqvist, R. Keskisärkkä, and E. Nuutila. Event processing in rdf. In *Proceedings of the 4th International Conference on Ontology and Semantic Web Patterns-Volume 1188*, 2013.
14. S. Sackmann, J. Strüker, and R. Accorsi. Personalization in privacy-aware highly dynamic systems. *Communications of the ACM*, 49(9), 2006.
15. B. Schneier and J. Kelsey. Cryptographic support for secure logs on untrusted machines. In *USENIX Security*, 1998.
16. O. Seneviratne and L. Kagal. Enabling privacy through transparency. In *Privacy, Security and Trust (PST), 2014 Twelfth Annual International Conference on*, 2014.
17. T. Waizenegger. Secure cryptographic deletion in the swift object store. In *Datenbanksysteme für Business, Technologie und Web (BTW)*, 2017.
18. T. Waizenegger, F. Wagner, and C. Mega. SDOS: using trusted platform modules for secure cryptographic deletion in the swift object store. In *Proceedings of the 20th International Conference on Extending Database Technology, EDBT*, 2017.
19. D. J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigenbaum, J. Hendler, and G. J. Sussman. Information accountability. *Communications of the ACM*, 51(6), 2008.
20. K. Wouters, K. Simoens, D. Lathouwers, and B. Preneel. Secure and privacy-friendly logging for egovernment services. In *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, 2008.
21. G. Zyskind, O. Nathan, et al. Decentralizing privacy: Using blockchain to protect personal data. In *Security and Privacy Workshops (SPW), 2015 IEEE*, 2015.