A Provenance-based Compliance Framework

Rocío Aldeco-Pérez, Luc Moreau

School of Electronics and Computer Science, University of Southampton,
Southampton SO17 1BJ, UK
{raap06r, l.moreau}@ecs.soton.ac.uk

Abstract. Given the significant amount of personal information available on the Web, verifying its correct use emerges as an important issue. When personal information is published, it should be later used under a set of usage policies. If these policies are not followed, sensitive data could be exposed and used against its owner. Under these circumstances, processing transparency is desirable since it allows users to decide whether information is used appropriately. It has been argued that data provenance can be used as the mechanism to underpin such a transparency. Thereby, if provenance of data is available, *processing becomes transparent* since the provenance of data can be analysed against usage policies to decide whether processing was performed in compliance with such policies. The aim of this paper is to present a Provenance-based Compliance Framework that uses provenance to verify the compliance of processing to predefined information usage policies. It consists of a provenance-based view of past processing of information, a representation of processing policies and a comparison stage in which the past processing is analysed against the processing policies. This paper also presents an implementation using a very common on-line activity: on-line shopping [1].

1 Introduction

Due to the increasing number of Internet services that manage sensitive personal information, the proper use of this information becomes a determinant issue for users who want to access these services with a guarantee that their personal information is not being misuse. Some research [14, 4, 8] is focused on developing better techniques to avoid information misuse by restricting access to information. However, access restriction alone cannot properly solve this problem on the Web, where information is widely and public available. When information becomes accessible, verifying its correct use after it was processed is also important. In order to solve this problem, users and organisations should be able to define policies under which personal information can be processed. In this context, information accountability, a property according to which users can analyse the way in which their information was used, is desirable. Weitzner *et al.* [16] and the W3C Provenance Incubator group [15] have argued that provenance, which consists of causal dependencies between data and events explaining what contributed to a result in a specific state [11], could be used to support information accountability to help users to answer questions related to the processing of information [16]. If provenance of data is available, processing becomes transparent since the provenance of data

[1] This research was partially supported by the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America, (E06D103956MX) and by the Mexican Council CONACyT (182546).

can be analysed against usage policies to decide whether processing was performed in compliance with such policies [1]. Information related to a specific processing can be obtained from provenance information by means of a provenance query [10], the result of which can be analysed to decide if the processing was performed in accordance with a set of usage policies.

In order to support this vision, we have created a provenance-based Compliance Framework that consists of a view of past information processing, a representation of the policies that processing should follow and a comparison stage in which the past processing is analysed against the processing policies. By using this framework, it is possible to decide if an application processed information in compliance to the predefined information usage policies. The framework components exploit provenance for representing both past execution and the rules to comply with. They are platform-independent and reusable, as they can be applied to different systems to verify different policies. They are also represented as an OPM [12] specialisation. In that way, any system using OPM can make use of this framework. At the same time, our framework could be used to create automatic auditing tools for verifying diverse policies over data processing.

The aim of this paper is to present this Compliance Framework; specifically, the contributions of this paper are: *(i)* The Compliance Framework components, which comprises the Processing View and the Usage Policies Definition, *(ii)* The Compliance Framework Analysis Stage, in which its components are compared to check the correct processing of information, and *(iii)* an implementation prototype.

The remainder of this paper is structured as follows. In §2, an application example is presented. In §3, the Compliance Framework's components are presented and explained. In §4, the framework's Analysis Stage and the algorithms used to verify some requirements are explained. In §5, an implementation of the previously explained example is presented. Finally, §6 discusses some related work and §7 offers some concluding remarks.

2 Application Example

Consider the following scenario: Alice is trying to get pregnant, and so has decided to take a fertility treatment (clomid). She decided to buy her treatment using the web page of a pharmacy. In order to get her treatment, she needs to provide her *name*, *address*, *date of birth*, *gender* and *social security number*. At the same time, but unrelated to her attempt to get pregnant, she applies for a job in the same pharmacy - and she is rejected. She suspects that the pharmacy may have checked its records related to her name and realised that she has plans to have a family, and as a result marked her as a high risk employee for expensive maternity costs. If this is true, the company obviously misused Alice's personal information. When she sent her personal information to the pharmacy, she did that with the purpose of getting her treatment. From the point of view of the company, the purpose is *on-line sales*. The on-line sales purpose could include verifying the existence and the sales of the medicine (manage stock), charging the amount to her card and sending the medicine to her home. The company can create a record of the monthly sales to manage medicines' stock. This record includes the *medicine's name* and the *quantity sold*. Nevertheless, such a record cannot contain the name of the people that bought that item, as the purpose of that record is not to identify

each person. By exposing the way in which the data sent by Alice was used, we can verify if the pharmacy correctly used her information and, if not, make it accountable. This example is represented in Figure 1 and analysed in the next sections to explain the framework.

3 Compliance Framework

The Compliance Framework consists of a past processing view, which is called *Processing View*, a policies representation, which is called *Usage Policies Definition*, and an analysis stage. The Processing View and the Usage Policies Definition are represented by provenance graphs that are directed acyclic graphs whose vertices represent data and edges relationships between such data. Data includes *purposes* (p_i), which are the intentions for which a set of data is to be collected, *tasks* (t_i), which are the processes performed over data, *data* that is to be collected (D_{Ci}) and processed (D_{Ii}), and *results* (r_i), which are the outputs of a task.

3.1 Processing View

The Processing View (PV) (Figure 1(a)) represents a provenance graph captured at execution time. We assume the entities involved in the processing of private information are capturing the corresponding provenance information following the approach presented in [1]. Also, to trust in the outcomes of the analysis stage, we assume that this view is secured in the way described in [2]. This view represents a process in which an application requests a set of data from a user, making explicit the purpose for which such a set **is acquired** (*collection purpose*). After checking the application purpose, the user send the requested multiset of data instances (*collected data*). The goal of the application is to achieve the collection purpose. For that reason, a *task* **is initiated by** the given purpose (*processing purpose*). Such a task **uses** a multiset of data instances that is a **subset of** the collected data (*used data*). Later, the task is executed with the used data as input and **generates** *results*. Such results could be used as collected data in the execution of another task. The *Processing View Graph* G_V can be defined as:

Definition 1 (Processing View Graph). *Let us consider a set of purposes P_V, a set of tasks T_V, a multiset of collected data instances D_{C_V}, a multiset of used data instances D_{U_V}, a multiset of results R_V and a set of relationship's names Rel_V =* $\{\mathsf{wasInitiatedBy}, \mathsf{used}, \mathsf{contained}, \mathsf{wasGeneratedBy}, \mathsf{wasAcquiredFor}\}$.

A Processing View Graph $G_V = (V_V, E_V, Rel_V)$ is a directed acyclic graph, where $V_V \subseteq P_V \cup T_V \cup D_{C_V} \cup D_{U_V} \cup R_V$, $E_V \subseteq (T_V \times P_V \times Rel_V) \cup (T_V \times D_{U_V} \times Rel_V) \cup (D_{U_V} \times D_{C_V} \times Rel_V) \cup (R_V \times T_V \times Rel_V) \cup (D_{C_V} \times P_V \times Rel_V) \cup (D_{U_V} \times R_V \times Rel_V)$.

3.2 Usage Policies Definition

The Usage Policies Definition (UPD) (Figure 1(b)) is a representation of the processing policies that are verified with the framework. These policies should be followed by applications while users' personal information is being processed. Each policy represents the way in which a set of data can be used, i.e. which tasks can use a certain multiset of

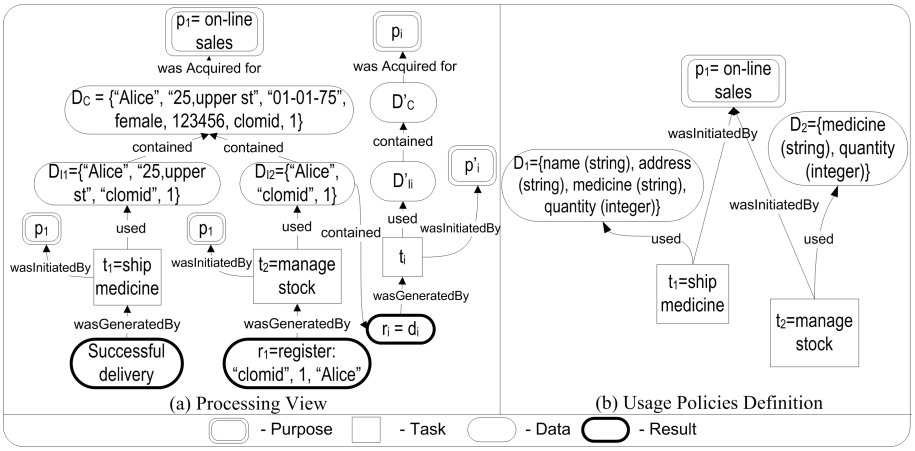


Fig. 1. Compliance Framework Components

data to accomplish which purpose. This definition contains a set of purposes from which users' data is collected, the set of tasks that **are initiated by** a specific purpose and a multiset of data types that a specific task **uses** in its execution. Note that this component contains more than one purpose and each purpose has more than one task. However, one task could be related to more than one purpose. The *Usage Policies Definition* G_R can be defined as:

Definition 2 (Usage Policies Definition Graph). *Let us consider a set of purposes* P_R*, a set of tasks* T_R*, a multiset of data types* D_R*, and a set of relationship's names* $Rel_R = \{\mathsf{wasInitiatedBy}, \mathsf{used}\}$.

A Usage Policies Definition Graph $G_R = (V_R, E_R, Rel_R)$ *is a directed acyclic graph, where* $V_R \subseteq P_R \cup T_R \cup D_R$, $E_R \subseteq (T_R \times P_R \times Rel_R) \cup (T_R \times D_R \times Rel_R)$.

4 Analysis Stage

In this stage, by using the already described components, it is possible to verify several information usage requirements, such as *(A) Purpose Compliance:* Processing of data is compatible with the purpose for which it was captured and *(B) Minimun Information Set:* Only information to be processed was captured. Due to space restrictions, we focus on *(A)*. The verification of these requirements is performed by comparing the past processing, described in the Processing View, against the expected processing, represented by the Usage Policies Definition. To do this, information related to the requirement is extracted from the Processing View in form of a subgraph. Then, the data and the relationships of such a subgraph are compared with the content of the Usage Policies Definition. The Purpose Compliance requirement states that the data used when performing a task should be data to only accomplish the initially stated purpose. The verification of this requirement includes to check if the correct type of data was used, if data was used to accomplish the stated valid purposes and, if data is reused, the processing is also made according to the stated purposes. From which, we derive the subrequirements *(1)*

Used Data Compliance, *(2) Purposes Validation* and *(3) Reusing Results*, respectively. Next, the extracted subgraphs related to each of these subrequirements are formally defined, the comparison process is explained presenting its corresponding algorithms and an example.

(1) Used Data Compliance Here, we verify that the used data has the correct data type related to the task that used it and the purpose that initiated such a task. We also verify that the purpose and the task are in the corresponding Usage Policies Definition. To this end, we extract from a Processing View Graph the provenance of a result. Specifically, we focus on the task that generated such a result, the data instances that such a task used and the purpose that initiated this task. This information is expressed as a subgraph of A_1, which is called *Used Data Compliance*.

Definition 3 (Used Data Compliance Subgraph (A_1)). *Given a Processing View Graph G_V, we can extract a Used Data Compliance Subgraph A_1 such that,*

$$V_{A_1} = \{p_i, D_{U_i}, t_i\}$$
$$E_{A_1} = \{(t_i, p_i, \mathsf{wasInitiatedBy}), (t_i, D_{U_i}, \mathsf{used})\}$$
where $p_i \in P_V, t_i \in T_V, D_{U_i} \in D_V \in R_V$ and $i =$ number of tasks.

Algorithm 1 Data Processed According to a Valid Purpose

input: $G_R = \{V_R, E_R, Rel_R\}, A_1 = \{V_{A_1}, E_{A_1}, Rel_{A_1}\}, p_i \in V_{A_1}, t_i \in V_{A_1}, D_{U_i} \subseteq V_{A_1}, D_j \subseteq V_R$
if $p_i \in V_R$ **then**
  if $t_i \in V_R$ **then**
    for all $x \in Rel_{A_1}$ and $y \in Rel_R$ **do**
      if $x = y$ **then return** -1 ▷ Label not matched
    for all $x \in D_{U_i}$ **do**
      if $type(x) \nsubseteq D_j$ **then return** -2 ▷ Type not matched
  else return -3 ▷ Not registered task
else return -4 ▷ Not registered purpose
return 1 ▷ Compliance

An example of this subgraph is presented in Figure 2(a). The UPD shows that the task "manage stock" should use a string (medicine's name) and an integer (medicine's quantity). However, as the PV depicts, this task used an extra data item: name. Therefore, the processing presented in Figure 2(a) does not satisfy the Used Data Compliance Subrequirement. The verification process of the Used Data Compliance subrequirement is presented in Algorithm 1 [2].

(2) Purposes Validation Here, we verify if the data was processed with a purpose related to the one it was collected for. To this end, we extract from a Processing View Graph the provenance of a set of tasks related to one set of collected data. Specifically,

[2] The type of each instance is included in the provenance information and obtained using the accessor $type$.

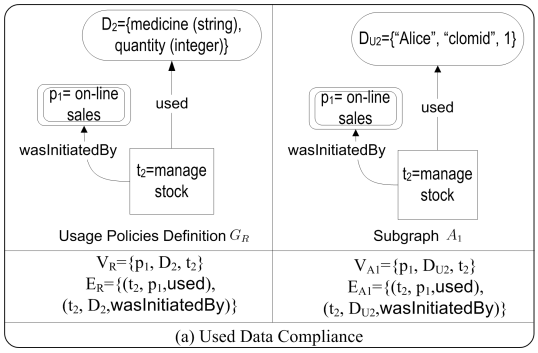


Fig. 2. Extracted Subgraphs

we focus on the purposes from which such tasks were initiated and the purposes from which the collected data was acquired. This is expressed in Figure 2(b) as the Purposes Validation subgraph, which is defined below.

Definition 4 (Purposes Validation Subgraph (A_2)). *Given a Processing View Graph G_V, we can extract a Purposes Validation Subgraph A_2 such that,*

$$V_{A_2} = \{p_i, D_C, D_{U_i}, t_i, p'_i\}$$
$$E_{A_2} = \{(t_i, p'_i, \mathsf{wasInitiatedBy}), (t_i, D_{U_i}, \mathsf{used}), (D_{U_i}, D_C, \mathsf{contained}), (D_C, p_i, \mathsf{wasAcquiredFor})\}$$
$$\textit{where } p_i, p'_i \in P_V, t_i \in T_V, D_{U_i} \in D_V, D_C \in D_V, i = \textit{number of purposes}$$

The Purposes Validation Subgraph is used to create a set of *Processing Purposes* (P'), which contains all p'_i, and a set of *Collection Purposes* (P''), which contains all p_i.

Algorithm 2 Processing Purposes vs Collection Purposes

```
input: G_R = {V_R, E_R, Rel_R}, A_2 = {V_A2, E_A2, Rel_A2}, P' = {p'_i}, P'' = {p_i}
if P' ⊆ P'' then
    if P' ⊆ V_R ∧ P'' ⊆ V_R then return 1        ▷ Compliance
    else return -1                               ▷ Not registered purpose
else return -2                                   ▷ Not compatible purpose
```

Later, we verify if the Processing Purposes are contained in the Collection Purposes, i.e. data was processed according to the purposes it was collected for. It is also verified if both sets contain valid purposes, i.e. they are contained in the UPD. The example presented in Figure 2(b) is in compliance with this subrequirement. The explained comparison process is presented in Algorithm 2.

(3) Reusing Data The previous requirements are used to verify the purposes related to data that was collected and processed by the same entity. However, an entity may

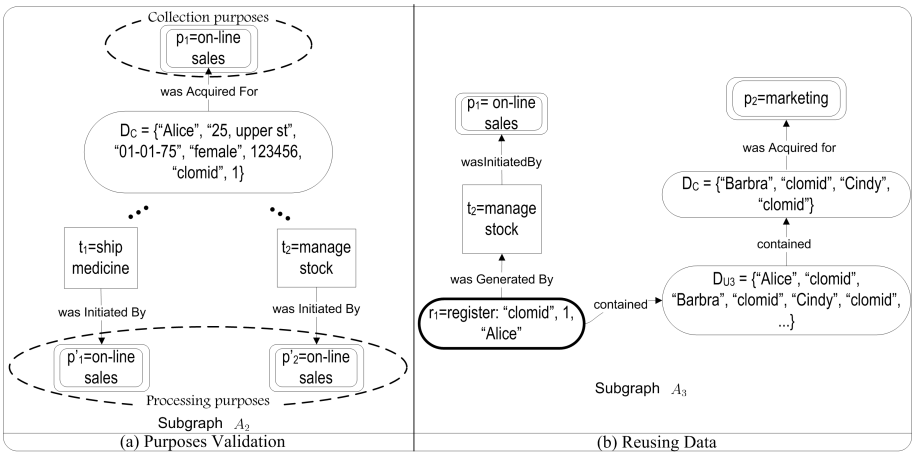


Fig. 3. Extracted Subgraphs

use data generated by tasks executed by other entities. This data is referred as "reused data" and it has its own processing purpose related to the task that produced it. To be in compliance to Requirement A, such a processing purpose should be related to the purposes for which data is reused. To this end, we extract from a Processing View Graph the provenance of the reused results that are the intersection of a collected data multiset related to one entity and a used data multiset related to one task executed by the same entity. Specifically, we focus on the task that generated such a reused data and the purpose from which such a task was initiated. This is expressed by the Reusing Data Subgraph of G_V, which is defined below.

Definition 5 (Reusing Data Subgraph (A_3)). *Given a Processing View Graph G_V, we can extract a Reusing Data Subgraph A_3 such that,*

$$V_{A_3} = \{R_j, t_j, p'_j\}$$
$$E_{A_3} = \{(R_j, t_j, \mathsf{wasInitiatedBy}), (t_j, p'_j, \mathsf{wasGeneratedBy})\}$$
where $p_j \in P_V, t_j \in T_V, R_j \in R_V$ and $j =$ number of reused data

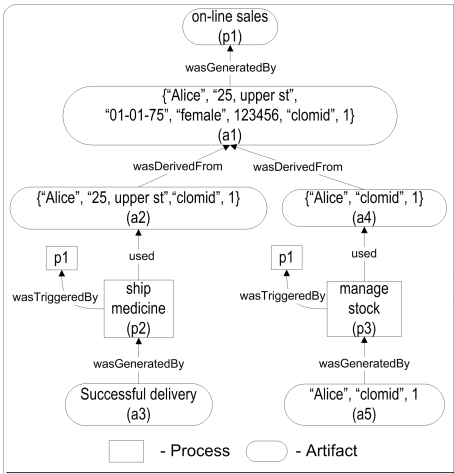
Algorithm 3 Reusing Data

input: $G_R = \{V_R, E_R, Rel_R\}, A_3 = \{V_{A_3}, E_{A_3}, Rel_{A_3}\}, P' \subseteq G_R, P'' \subseteq G_R, p \in V_{A_3}$
if $p \in P'$ and $p \in P''$ **then return** 1 ▷ Compatible purpose
else return -1 ▷ Not compatible purpose

In the example presented in Figure 3(a), the reusing of the result of the task "manage stock" is presented. Some data items of this result are used in a new task with purpose "marketing". However, the result was initiated by the purpose "on-line sale". Therefore, reusing this result is not in compliance with the initial purpose. The verification process of this subrequirement is presented in Algorithm 3.

5 Implementation

To show how the Provenance-based Compliance Framework is used, we implement the previously presented example. Due to the lack of space, we focus on the queries used to obtain the information we need to verify the subrequirement "Used Data Compliance". To this end, the processing view graph, which is presented in Figure 4(a), is represented as an RDF OPM Graph [12] and the queries are implemented using SPARQL. Initially,



(a) Example Processing View

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX opm: <http://openprovenance.org/ontology#>
PREFIX ex: <http://example.com/>
SELECT DISTINCT ?alabel WHERE {
 ex:gr_14 opm:hasDependency ?d.
 ?d ?cep ?ce;
    a opm:Used.
 ?ce opm:label ?label.
 FILTER (((?d opm:cause ? ce = "http://openprovenance.org/
ontology#cause") ||(?d opm:effect ? ce = "http://
openprovenance.org/ontology#effect")) && ?label = "ship
medicine").
 ?d ?da ?a.
 ?a a opm:Artifact;
    opm:label ?alabel}
```

(b) Obtain used data from the ship medicine task

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX opm: <http://openprovenance.org/ontology#>
PREFIX ex: <http://example.com/>
SELECT DISTINCT   ?d ?ddp ?dv WHERE {
 ex:gr_14 opm:hasDependency ?d.
 ?d opm:effect ex:p2.
 ?d a opm:WasTriggeredBy.
 ?d ?ddp ?dv}
```

(c) Obtain purpose that triggered the ship medicine task

Fig. 4. Implementation Example

we verify the *ship medicine* task. To this end, we extract the used data (D_U) using the SPARQL query presented in Figure 4(b). We also need to extract the purpose (p) that initiated the *ship medicine* task by executing the query presented in 4(c). The task name (t) is extracted by using a similar query, which is not shown for the given space restrictions. Then, the obtained information (D_U, p, t) along with the corresponding Usage Policies Definition, are used as the input of Algorithm 1. After executing this algorithm, the result is that the task *ship medicine* is in compliance with the given requirement, so we can say that the processing of Alice's data by this task was in compliance. The task *manage stock* can be analysed in a similar way. If we extract the necessary elements related to this task and apply Algorithm 1, the result will be no compliance. The reason is that a piece of data that is not part of the policies has been used: *name*. An inventory of a pharmacy does not need the name of the customers, just the item that was sold and the quantity of it. Then, we can say that the processing of Alice's information by this task was not in compliance, and therefore, such an inventory could be used against her interest. In this case, to be marked as a high risk employee.

6 Related Work

Recently, researchers have realised that provenance can be used to verify the compliance of different policies related to the use of personal information [6, 5, 13] and others use different technologies to support compliance [3, 9, 7].

Hanson *et al.*[6] present a data-purpose algebra to annotate data with usage restrictions. With these restrictions, requirements similar to the ones presented here can be checked. In this approach, each data is annotated after execution, contrary to our approach, where provenance information created at execution time is used. In terms of implementation, they just present a prototype to verify requirements compliance. Gil *et al.* [5] argue that computational workflow systems can be used to ensure and enforce the appropriate use of sensitive personal data. Then, compliance, transparency and accountability can be supported by these systems. To this end, they define a set of requirements that are similar to the ones we present here. Later, using workflow system to support process transparency, the defined requirements can be verified. They also propose to use this technology to enforce policies and negotiate them. In our work, we propose that given the openness of the Web to support compliance is more flexible than enforcement. Finally, they do not offer a practical solution as the one presented in this paper. Ringelstein *et al.* [13] use a modified version of OPM to express processing execution and an extended version of XACML to represent permission and restriction policies. Then, by modelling conditions, enforcement is supported. As we previously mention, we support accountability, not enforcement. We also create OPM-based rules, which can be easily checked against any OPM-based view without any previous transformations.

In the business processes context, Ly *et al.*[9] present a similar work to check the compliance of rules related to the quality of a product. They requirements are different as they verify the executed order of processes. They use process-aware information systems and process models to model the rules. Awad *et al.* [3] check compliance of control flow and data flow in business process models by using BPMN to express process models and BPMN-Q to represent policies. They design a set of queries and present how violations occur by comparing predefined patterns with the queries. These two approaches do not support the verification of use of data items and are not based in an open model, such as OPM. Kang *et al.* [7] also present a similar approach that helps users to conform with existing policies in a social networks context. This is achieved by making them aware of data usage restrictions defined by the users. One drawback of this work is that they just define a small set of policies, contrary to our framework, in which any usage policy can be defined. In that way, our framework can be applied to a diversity of contexts.

There is also work on authorisation and enforcement of rules in databases [14, 4, 8]. However, given the openness of information on the Web and the possibility of inferring information using previously published information, authorisation and enforcement are very challenging to implement.

7 Conclusions

In this paper, we present the provenance-based Compliance Framework by explaining its components and how this framework can be applied to an application example over

an on-line shopping scenario. With this example, we explain how our framework helps us verify one (of more) requirement related to the processing of information. Our work demonstrates that by using the Provenance-based Compliance Framework, individuals or institutions, which used information in a different manner from the stated, can be held *accountable* for misuse. Our framework comprises one past processing view, one novel policies definition, which are platform independent and reusable, and one comparison stage, which is easy to implement in an automatic way.

References

1. R. Aldeco-Pérez and L. Moreau. Provenance-based Auditing of Private Data Use. In *International Academic Research Conference, Visions of Computer Science*, September 2008.
2. R. Aldeco-Pérez and L. Moreau. Securing Provenance-based Audits. In *IPAW '10 (In Press)*, Troy, NY, 2010.
3. A. Awad, M. Weidlich, and M. Weske. Specification, Verification and Explanation of Violation for Data Aware Compliance Rules. In *Lecture Notes In Computer Science; Vol. 5900*, volume 5900, pages 500–515, Stockholm, 2009. Springer-Verlag.
4. A. Cirillo, R. Jagadeesan, C. Pitcher, and J. Riely. TAPIDO: Trust and Authorization via Provenance and Integrity in Distributed Objects , 2008.
5. Y. Gil and C. Fritz. Reasoning about the Appropriate Use of Private Data through Computational Workflows. In *AAAI Spring Symposium on Privacy Management 2010*, pages 23–25, 2010.
6. C. Hanson, T. Berners-Lee, L. Kagal, G. J. Sussman, and D. Weitzner. Data-Purpose Algebra: Modeling Data Usage Policies. In *IEEE Policies for Distributed Systems and Networks*, pages 173–177, Bologna, Italy, May 2007. IEEE.
7. T. Kang and L. Kagal. Enabling Privacy-awareness in Social Networks. In *Intelligent Information Privacy Management Symposium at the AAAI Spring Symposium 2010*, 2010.
8. W. Lu and G. Miklau. Auditing a Database under Retention Restrictions. *ICDE*, pages 42–53, 2009.
9. L. T. Ly, S. Rinderle-Ma, and P. Dadam. Design and Verification of Instantiable Compliance Rule Graphs in Process-Aware Information Systems. In *22nd Int'l Conf. on Advanced Information Systems Engineering (CAiSE'10)*, 2010.
10. S. Miles. Electronically querying for the provenance of entities. In *Proceedings of the International Provenance and Annotation Workshop IPAW*, pages 184–192. Springer, November 2006.
11. S. Miles, P. Groth, S. Munroe, S. Jiang, T. Assandri, and L. Moreau. Extracting Causal Graphs from an Open Provenance Data Model. *Concurrency and Computation: Practice and Experience*, 20(5):577–586, Apr. 2007.
12. L. Moreau, B. Clifford, J. Freire, Y. Gil, J. Futrelle, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, Y. Simmhan, E. Stephan, J. Van Den Bussche, and B. Pale. The Open Provenance Model Core Specification (v1.1). *Future Generation Computer Systems*, pages 1–30, 2010.
13. C. Ringelstein and S. Staab. PAPEL: A Language and Model for Provenance-Aware Policy Definition and Execution. In *8th International Business Process Management Conference*, 2010.
14. J. A. Vaughan, L. Jia, K. Mazurak, and S. Zdancewic. Evidence-based audit. In *21th IEEE Computer Security Foundations Symposium*, pages 177–191. IEEE Computer Society, 2008.
15. W3C. Provenance incubator group, October 2009.
16. D. J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigenbaum, J. Hendler, and G. J. Sussman. Information accountability. *Communications of the ACM*, 51(6):82–87, 2008.