EURECOM
Department of Network and Security
Campus SophiaTech
CS 50193
06904 Sophia Antipolis cedex
FRANCE

Research Report RR-14-294

A-PPL: An Accountability Policy Language

August 11th, 2014

Monir Azraoui, Kaoutar Elkhiyaoui, Melek Önen*, Karin Bernsmed†, Anderson Santana de Oliveira and Jakub Sendor‡
*EURECOM, †SINTEF ICT and ‡SAP Labs France

Tel : (+33) 4 93 00 81 00
Fax : (+33) 4 93 00 82 00
Email : *{monir.azraoui, kaoutar.elkhiyaoui, melek.onen}@eurecom.fr
†karin.bernsmed@sintef.no
‡{anderson.santana.de.oliveira, jakub.sendor}sap.com

[1] EURECOM's research is partially supported by its industrial members: BMW Group Research and Technology, IABG, Monaco Telecom, Orange, Principaut de Monaco, SAP, SFR, ST Microelectronics, Symantec.

A-PPL: An Accountability Policy Language

Monir Azraoui, Kaoutar Elkhiyaoui, Melek Önen*, Karin Bernsmed†, Anderson Santana de Oliveira and Jakub Sendor‡
*EURECOM, †SINTEF ICT and ‡SAP Labs France

Abstract

The inherent lack of control of users over their data raises various security and privacy challenges in Cloud Computing. One approach to encourage customers to take advantage of the Cloud is the design of new accountability solutions which aid and enable customers to control and be informed on how their data is processed. In this paper, we focus on accountability policies and propose A-PPL, an accountability policy language that represents machine-readable accountability policies. A-PPL policies provide cloud customers and cloud end-users with a way to express accountable obligations in order to automate their enforcement. Our work also describes a use case where medical sensors collect personal data which are then stored and processed in the cloud. We define the accountability obligations related to this use case and translate them into A-PPL policies as a proof of concept of our proposal.

Index Terms

Cloud Computing, Accountability, Policy Language and Enforcement

Contents

List of Figures

List of Tables

1 Introduction

Cloud computing marks a shift in the way organizations and individuals consume technology. The term congregates a number of delivery models where computing infrastructure (CPU, memory, network), platform (databases, middleware, application servers etc.) or software is provided as a service, offering scalability and reducing capital expenditure thanks to the elastic resource allocation. In the cloud computing paradigm, cloud customers delegate the implementation of numerous security and privacy controls to the cloud service provider (CSP), raising accountability concerns. In particular, business customers perceive data lock-in, loss of governance and non-compliance as major risks associated with the cloud [1].

Organizational cloud customers usually assume the role of data controller, thus they are held accountable for the way cloud services respond to many regulations, including the EU Data Protection Directive [2]. Cloud customers nowadays lack means of control on how data is processed in the cloud, therefore, they are conferring a higher level of trust onto the CSPs when compared to the actual guarantees the customers obtain. Cloud services are typically offered in standard form contracts and agreements. Such agreements may not explicitly address in which way obligations regarding personal data are carried out, as they are often drafted by providers and not customers [3].

Clarifying the accountability relationships, i.e. who is responsible to whom and for what, help overcome barriers to data governance in the cloud. As defined in [4], accountability concerns data stewardship regimes in which organizations that are entrusted with personal and business confidential data are responsible and liable for processing, sharing, storing and otherwise using the data according to contractual and legal requirements from the time the data is collected until when it is destroyed (including onward transfer to and from third-parties). In such a setting, clear organizational policies are a fundamental requirement for control mechanisms. Appropriate policies mitigate risks, provided that reliable tools to enforce them and to monitor their effectiveness are in place to allow audits.

In this work we are interested in machine-readable representations of policies expressing accountability obligations. Such policies will help service providers deploy automatic enforcement of privacy and security policies when they process personal data. We design a new policy language that enables the expression of the accountability obligations.

Contributions. We outline here our contributions:

1. This paper presents a number of accountability obligations from which we derive the requirements for an accountable policy language.

2. We build upon the PPL work [5, 6] this new language called A-PPL (shorthand for Accountable-PPL). We also outline the components of the A-PPL engine which takes care of the enforcement of A-PPL rules.

3. We finally validate our accountability policy language by modeling a use case. We show how to translate into A-PPL policies the obligations extracted from this use case.

The remainder of the paper is structured as follows: Section 2 introduces accountability obligations coming from different perspectives and formulates from these identified obligations the design requirements for an accountable policy language. Section 3 gives an analysis of existing policy languages with respect to our requirements. We describe then our language A-PPL in Section 4. Section 5 presents the use case based on medical sensor networks in the cloud from which we define accountability obligations that we translate into A-PPL statements. We review the related work in Section 6. Conclusion and future work are part of Section 7.

2 Accountability Obligations and Policy Language Requirements

Accountability obligations may derive from regulations (for example, the European data protection law [2]) and contractual agreements (SLA, Terms of Use, privacy policies, etc.). In this section we define the actors providing and processing personal data and we identify from their relationships a set of accountability obligations. By analyzing these obligations, we define the requirements that guided our design of a machine-readable policy language for accountability.

2.1 Accountability Obligations

Accountability obligations concern the relationships among the following actors:

Data controllers are “the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes and means of the processing of personal data” [2]. Organizations that purchase cloud services are often controllers.

Data subjects are the individuals from whom personal data is collected. Data subjects are often the end users of a cloud service.

Data processor is the natural or legal person, public authority, agency or any other body which processes personal data on behalf of the controller. Cloud providers will become processors when their customers use their services to process personal data.

Data Protection Authorities (DPA) represent national supervisory authorities, such as the Information Commissioner’s Office (UK), the French CNIL, the German BFDI, etc.

The relationships between these actors define accountability obligations that have to be fulfilled according to regulations and contracts.

• Data controllers are accountable to data subjects for:

The right to information: Data subjects have the right to know that their personal data is processed and for which purpose.

Data quality: Data quality means that personal data must be, for instance, processed fairly and lawfully, collected for specified, explicit and legitimate purposes, and kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the data was collected or further processed [2].

Confidentiality: This is the obligation of any person acting under the authority of the controller or the processor, and who has access to personal data, not to process them except on instructions from the controller, unless he is required to do so by law.

• Data controllers are accountable to DPA for:

Notification on processing operations of personal data: The data controller must explain the context of the personal data processing and justify the purposes of the processing. Accountability policies for the cloud can clarify roles and responsibilities, allowing for auditable enforcement of privacy constraints.

International data transfers (change of data location): Some international legal mechanisms frame personal data transfers across countries, for instance, Binding Corporate Rules (BCRs). The data controller is hence accountable for obtaining authorization from the DPA for international transfers.

The assignment of processing operations to data processors: Data controllers are accountable to DPAs for choosing those data processors that provide sufficient safeguards concerning the technical security and the organizational measures required in relation to the processing to be carried out on their behalf.

• Data processors are accountable to data controllers for:

Contractual obligations: This means that they are required to provide the service as specified in the contracts.

Confidentiality/Security control obligations/Data integrity: These requirements relate to regulatory obligations of data security, breach, data loss and confidentiality, etc.

Data location: The objects stored in a region must never leave the region unless the customer transfers them out.

Besides, the above obligations highlight the needs for collecting evidence on the cloud service operations and implemented security controls. For instance, audits from DPAs may require the collection of logs that record the actions performed by data controllers and processors.

Accountability policies will be particularly useful for specifying concrete obligations in cases where data controllers outsource the processing of personal data to cloud providers. A machine-readable policy language can convey these accountability policies. Our goal is to design an accountability policy language that eases and automates their enforcement. Therefore we derive from the analysis of the accountability obligations several design requirements for our policy language.

2.2 Policy Language Requirements

Table 1: Accountability policy language requirements.

Requirement	**Category**
(R1) Capturing Privacy Policies	Data Handling
(R2) Access Control Rules	Data Handling
(R3) Usage Control Rules	Data Handling
(R4) Data Retention Period	Data Handling
(R5) Reporting and Notification	Accountability
(R6) Controlling Data Location	Accountability
(R7) Auditability	Accountability
(R8) Logging	Accountability

Table 1 summarizes our analysis of regulatory obligations concerning personal data processing. We classify requirements as either data handling or accountability requirements. The former refer to the need to express privacy constraints, access and usage control rules. The latter correspond to the requirements that are specific to accountability and which are often not addressed by existing policy languages such as audits, logging and notifications

In this section we explain the requirements that we identify based on the obligations described in section 2.1.

- **Data handling requirements:**

(R1) Capturing privacy policies: Our accountability policy language must allow the expression of privacy policies about the usage of personal data.

(R2) Access Control Rules: We identify the obligation of confidentiality of data. Therefore, our accountability policy language must enable the specification of access control policies to personal data. The access requester should in particular be defined by a set of attributes such as its name, its role, or the group it belongs to.

(R3) Usage Control Rules: The obligation on data quality suggests the definition of appropriate usage control rules. Our accountability policy language must allow the expression of such rules. In particular, it should express the conditions under which an action on the data is permitted or prohibited (such as sharing the data with third parties, usage for a particular purpose). It should also define the operations on the data that has to be performed after its collection (such as deletion, anonymization, etc.).

(R4) Data Retention Period: Data quality also deals with data retention periods. Our accountability language must be able to express rules about data (and metadata) retention such as retention time.

- **Accountability requirements:**

(R5) Reporting and Notification: Data subjects, cloud users or DPAs should be able to receive notifications about the usage and the processing of personal data, about security breaches or about policy violations. The policy language we design should enable the sending of notifications to data subjects and third parties.

(R6) Controlling Data Location: As controlling data location is an obligation for which data controllers are accountable to data subjects and DPAs, the language must enable the expression of rules about data location in a policy.

(R7) Auditability: Accountable services may be audited to verify compliance with obligations. Therefore, our accountability policy language must make possible the auditing of operations performed in the cloud (such as deletion, transfer, modification, access, etc.). The language must also specify what information is targeted by an audit, and which evidence should be collected to perform the audit.

(R8) Logging: Evidence collection is one of the obligations we identified above. Logs can be a particular type of evidence. Therefore, the policy language must specify which events have to be logged and what information related to the logged event have to be added in the log.

One may argue that these requirements can be expressed and enforced using multiple languages at different levels of the cloud technology stack. We advocate that centralizing these concerns in a single policy will increase the accountability of the actors processing personal data in the cloud, while decreasing the loss of governance, as policies will not be diluted across the service provisioning chain.

We use these requirements to review and analyze existing policy languages so as to design suitable accountability policy language. The results of this review is presented in Section 3.

3 Background on Policy Languages

We aim at determining to which extent existing policy languages meet the requirements defined in Section 2.2. We select a total of nine policy languages. The review of the state of the art takes into account existing standards and languages that allow to define machine-readable policies for access and usage control, privacy and contract negotiation. We analyze in particular the ability of these languages to map accountability obligations and the possibility to extend these frameworks for such a mapping. Rather than imposing a new language for expressing existing security and privacy obligations, we aim at choosing the existing language which expresses the best the accountability obligations and which is extensible enough to add accountability extensions to it. This analysis is presented in Appendix A.

3.1 PrimeLife Policy Language (PPL)

As a result of this review, we focus our effort on the extension of PPL.

PPL [5] was proposed by the European ICT PrimeLife[1] project. The focus of the language is to enforce by technical measures privacy statements written in natural language. In particular, it helps write machine-readable policies about the handling of personal data and its forwarding to third-parties called *downstream* users. PPL extends XACML [7] with the following items.

- *a new obligation syntax.* In PPL, an obligation is expressed using the pair Trigger-Action. Triggers are events related to an obligation and filtered by conditions. For example, PPL defines the trigger `TriggerPersonalDataDeleted` that occurs whenever the personal data related to the obligation is deleted. **Triggers** fire **Actions** that are performed by the data controller. For instance, PPL provides the action `ActionNotifyDataSubject`. The complete list of available PPL Triggers And actions can be found in the appendix B.

- *an authorization language,* that defines the actions that the data controller is allowed or prohibited to perform: (i) authorization for purposes, allows the data controller to perform actions for a particular set of well-stated usage purposes; (ii) authorization for downstream usage, allows the forwarding of collected information to third parties (downstream data controllers) under a particular privacy policy.

PPL is a good candidate language to express accountability obligations. Indeed, the language fulfills most of the requirements we identify in Section 2. PPL

[1] http://www.primelife.eu/

allows the data controller to write privacy policies (R1). Access and Usage control rules are the aim of PPL. Thus, requirements (R2) and (R3) are fulfilled in PPL. Data retention periods (R4) can be specified in PPL using a specific combination of action and trigger: `ActionDeletePersonalData` triggered by a temporal trigger `TriggerOnTime` that specifies the data retention period.

3.2 Limitations of PPL for accountability

PPL presents some limitations and does not accurately meet the other language requirements. PPL enables reports and notification (R5): the current specification of PPL defines the action `ActionNotifyDataSubject`. However, it may not be useful when the recipient of a notification is someone else than the data subject. In addition, this PPL action does not inform about the type of notification to be sent. In an accountable cloud environment, notifications can be of several types (policy violation report, redress measures notification, logs, etc.). Furthermore, the current specification of PPL can declare the action `ActionLog`, to log an event based on a trigger. This partially covers the requirement for logging (R8). However, the information that has to be put in the log is not part of the PPL element. Besides, controlling data location (R6) and auditability (R7) are not part of PPL. PPL provides no way to request and perform an audit, to handle the collection of evidence that has to be presented for the verification of compliance with policies, user preferences or regulations. Moreover, there is no way in PPL to specify the location of the data. Neither the data controller, nor the data subject can express their policies or preferences on where the data has to be kept. Finally, the obligations related to logging, evidence collection and auditing are not part of the PPL engine which aims at enforcing PPL policies.

Having identified the limitations of PPL, we propose Section 4 our accountable policy language A-PPL that extends PPL.

4 A-PPL: Accountable Policy Language

In this section we present the extensions we add to PPL to create A-PPL. Note that we maintain the overall structure of PPL.

4.1 A-PPL language as an extension of PPL

4.1.1 Roles

To make the identification of roles more explicit in an accountable cloud, we include in a policy a reference to the role of the different entities involved in the policy. These roles are those identified in Section 2.1. Thus, we create a role attribute identifier `subject:role` to be included as an attribute of the standard XACML element <`Subject`>. In addition, we propose to define the role of the auditor in A-PPL. This new role is useful for accountability specific obligations such as reporting and notification (R5) or auditability (R7).

4.1.2 Access Control Rules (R2)

We introduce two new triggers which condition the execution of an obligation based on the result of an access decision. In other words, we propose `TriggerPersonalDataAccessPermitted` and `TriggerPersonalDataAccessDenied` that occur when the evaluation of the access control on the targeted data results in “Permit”, respectively “Deny”.

4.1.3 Data Retention (R4)

PPL provides an element `Purpose` that allows to specify for which purpose a piece of data can be collected or accessed. In A-PPL, we define the `duration` attribute for `Purpose` that allows to specify for how long the data can be processes for a particular purpose. For instance, a particular piece of data is used for research purposes for 2 years but has to be kept for legal purposes for 5 years. In addition, this attribute implies that when all durations for each purpose have expired, the data has to be deleted, since the data cannot be used for any purpose anymore.

4.1.4 Reporting and Notification (R5)

We modify the existing PPL `ActionNotifyDataSubject` element and call the newly created notify action `ActionNotify`. Notifications are not limited to notifications to the data subject only. Instead, we provide an attribute `recipient` that allows to indicate the recipient of the notification. The `ActionNotify` element presents an attribute `type` that specifies the type of notification to be sent to the recipient (policy violation report, audit reports, etc.). Table 2 describes the `ActionNotify` element.

Table 2: ActionNotify element.

Name	**ActionNotify**	
Description	This action notifies a cloud actor when triggered	
Parameters	Media	The media used to notify the user (e-mail, SMS, etc.)
	Address	The corresponding address (e-mail address, phone number, etc.)
	Recipient	The identity of the recipient of the notification
	Type	The type of notification(policy violation, evidence, redress, etc.)

4.1.5 Controlling Data Location (R6)

We propose in A-PPL a standard identifier `environment:region` for such environment attribute. It will be used in the XACML policy target <`Environment`>

element of an A-PPL policy to specify the location of the subject who can obtain the access to the data. Thus we will limit the region among which the data can be transferred without violating the policy access control rules. This is directly responding to our requirement on controlling data location in the policy language (R6).

4.1.6 Auditability (R7)

We propose two extensions that relate to audits and collection of evidence. Based on the evidence request that the auditee receives from the auditor, the auditee collects the requested evidence. This evidence collection is governed by a new A-PPL trigger `TriggerOnEvidenceRequestReceived`, and a new A-PPL action `ActionEvidenceCollection`. The combination of the two A-PPL elements initiates the evidence collection by the data controller. Table 3 describes the `ActionEvidenceCollection` element.

Table 3: ActionEvidenceCollection element.

Name	**ActionEvidenceCollection**	
Description	This action collects the requested evidence	
Parameters	Evidence	The type of evidence to generate (logs, crypto proofs, etc)
	Resource	The ID of the resource the evidence is based on
	Subject	The ID of the data subject the evidence is based on
	Recipient	The ID of the recipient of the evidence (the auditor)

4.1.7 Logging (R8)

We extend the `ActionLog` element in A-PPL. In particular, we introduce several parameters to make explicit which information about an event needs to be logged. A timestamp is required to log the time of the event. The policy must indicate to log the action that is performed on the data (e.g. `SEND`), the identity of the subject who performed the action (e.g. `Cloud x`) and the purpose of the action (e.g. `marketing`). To trace events based on data, the policy must require the identifier of the data. Other details must also be written in the logs such as some security flags that may state whether the log entry is encrypted. Table 4 describes the `ActionLog` element.

We also propose additional extensions such as `TriggerOnPolicyUpdate` or `TriggerOnComplaint`. For a more comprehensive description of these extensions, the reader may refer to [8].

4.2 A-PPLE: the extension of the PPL engine

Privacy policy engine supporting PPL was originally designed in PrimeLife project [5]. We adapt its architecture to implement the new requirements about

Table 4: ActionLog element.

Name	**ActionLog**	
Description	This action logs an event based on the details in the policy	
Parameters	Timestamp	The time of occurrence of the logged event
	Action	The action that is logged
	Purpose	The purpose of the action that is logged
	Subject ID	The identity of the subject that performed the action
	Resource ID	The identifier of the resource the action was made on
	Resource Location	The location of the resource
	Security Flag	1 if the log is confidential, 2 for integrity check, 3 for both

accoutability, creating the architecture depicted in Figure 1. The engine supports the enforcement of data handling constraints (usage and access control) stated in the accountability policy.

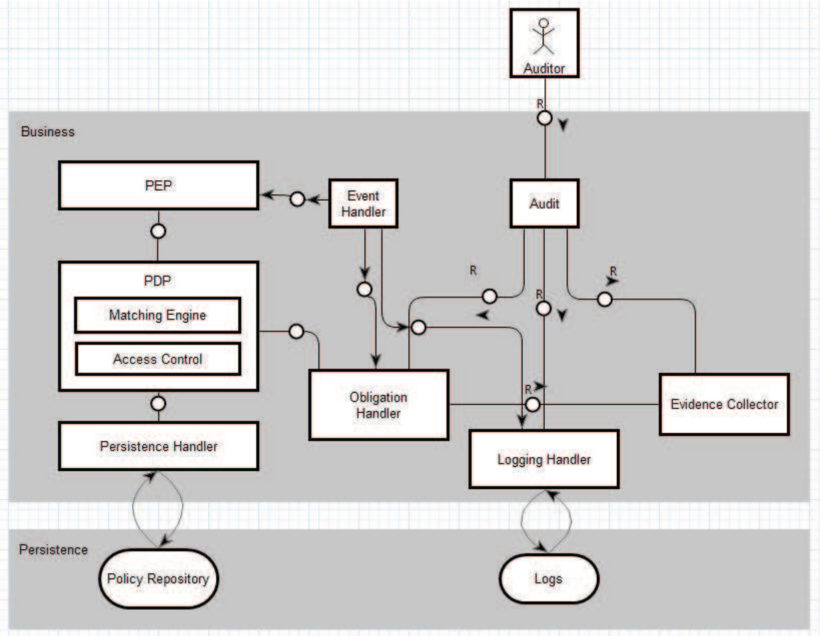

Figure 1: A-PPL Engine architecture

The core elements of the policy engine are the components in the Business Layer: Policy Enforcement Point (PEP) and Policy Decision Point (PDP). While the PEP acts as an orchestrator of the enforcement process and interface with the Web Services, the PDP is the component where the access control decision is taken.

PDP relies on the access control engine implementation based on HERAS [9] for the evaluation of XACML part of PPL policy. Apart from the standard attribute-based access control, the other information evaluated by the PDP at the step of access control decision is usage authorization and the result of policy matching. The usage authorization basically consists of the comparison of the list of purposes specified in the data subject preferences with the one specified by data controller in his policy. It also compares the authorization for the downstream usage (use of the data by the third parties, with whom data controller might share the collected data in the future).

As the personal data is stored together with the associated privacy policy in the Database, the PDP communicates with the PII Store/Policy-Preference Store by the Persistence Handler interface.

The PEP coordinates two modules: the Event and Obligation Handlers. The functionality of the Event Handler is to fire the events related to the personal data lifecycle, e.g. when data is deleted from the PII store or when it is shared with the third parties.

The Obligation Handler keeps track of the triggers that are part of the obligation statements in the A-PPL policy. It is initialized after new policies are entered to the repository and updated every time a new PII data item is created. Once the events are observed, which might be the case of receiving the notification from the Event Handler for the event-based triggers or simple time-outs in the case of the time-based triggers, the action associated with the obligation is activated by the Obligation Engine.

We add a central component for handling the audit requests, which will facilitate the process of retrieving the necessary information from the systems (logs related to obligations, notifications, access control decisions and personal data lifecycle). Furthermore, each component in engine architecture that is related to this information (Obligation Handler, Event Handler, PDP and PEP) are linked to the logging adapter making it possible to record all data sensitive actions in a non-repudiable manner.

5 Validation of A-PPL with a use case

In this section we present a use case that illustrates how accountability obligations can be expressed using A-PPL. We first introduce the use case that deals with the flow of healthcare information collected by medical sensors. We then extract accountability obligations from the use case. Finally, we show how A-PPL can be used to address these obligations.

5.1 Use case: Medical sensor networks in the Cloud

The use case that we describe is a healthcare system that will be used to support elderly people by analysis of medical data collected by sensors. We investigate a case where medical data from the sensors will be exchanged between the elderly, their families and friends, hospital caregivers and healthcare personnel. The pro-

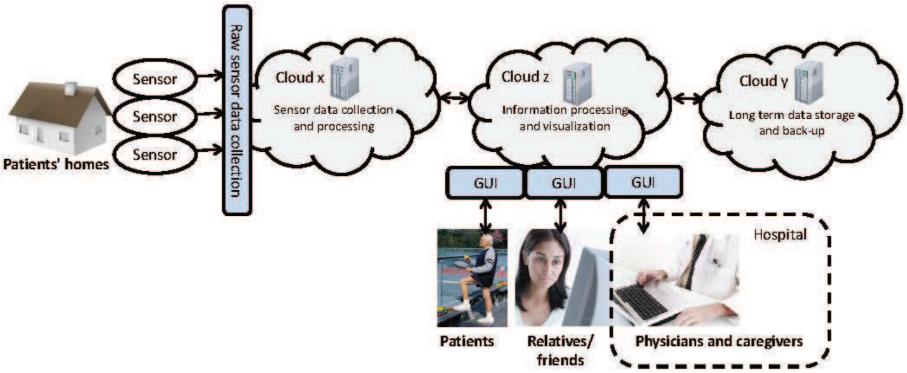
Patients' homes
Sensor
Sensor
Sensor
Raw sensor data collection
Cloud x
Sensor data collection and processing
Cloud z
Information processing and visualization
Cloud y
Long term data storage and back-up
GUI
GUI
GUI
Hospital
Patients
Relatives/ friends
Physicians and caregivers

Figure 2: An overview over the healthcare use case

posed solution is the M Platform illustrated in Figure 2, which is a cloud-based service for medical sensor data collection, processing, storage and visualization. Patients will be connected to wireless sensors that monitor their vital signs (e.g., movement, blood pressure, pulse oximetry, temperature, position, etc.). The sensor data will be transmitted to the cloud where they will be further processed and stored.

The M Platform is offered to the hospital as a service from a European software and service provider M, which has outsourced both the initial storage of data collected through the sensors placed by hospital staff (Cloud x, which is provided by X) as well as the long-term data storage and back-up procedures (Cloud y, which is provided by Y). Note that the providers of Cloud x and Cloud y are engaged by M; the hospital has no direct contractual relationship with either X or Y. The information engine, which visualizes and displays information to the end users, is a cloud service which will be implemented by M in M's own infrastructure (Cloud z). The Cloud z service integrates and communicates with Cloud x (and Cloud y, separately). As can be seen in Figure 1, through graphical user interfaces (GUIs) the M Platform will interact with and provide services to a number of different users involved. In this use case, sensors communicating with the M Platform are used to collect sensor data from elderly persons who are suffering from dizziness, in order to help make a diagnosis.

In this use case the patients are the data subjects, since they are the individuals, from whom the system collects personal data, including location, blood pressure, oxygen saturation, and more. The hospital is a controller of its patients' personal data which it has chosen to process using M's cloud service. M (who is the primary service provider to the hospital) is the hospital's processor of the patients' personal data. Also the cloud sub-providers X and Y, which have been engaged by M, are processors[2].

[2] Also relatives/friends and hospital staff will be data subjects, and under certain circumstances also controllers (w.r.t. the patients' personal data).

5.2 Obligations derived from the use case

To comply with the European Data Protection Directive, as well as with the contractual relationships that must exist between the involved actors, a number of accountability obligations can be derived for the healthcare use case. Here we outline some of the more prominent ones. Further details can be found in the paper by Bernsmed, Hon and Millard [10]. For each obligation, we also provide a way to map them into A-PPL statements.

Obligation 1: The right to access, correct and delete personal data The hospital must ensure that the patients have read and write access to their personal data that have been collected and stored in the cloud. There must be also means to enforce the deletion of such data.The right to access is expressed in XACML rules that A-PPL is built upon. The data controller grants both read and write access to the data subject. In addition, the deletion of the personal data can be ruled by an A-PPL data handling policy whereby the obligation to delete the data can be expressed using the A-PPL `ActionDeletePersonalData` in conjunction with the trigger `TriggerAtTime`.

Obligation 2: Purpose of processing The hospital must make sure that the patients' personal data is only processed for specific, explicit and legitimate purposes. A-PPL uses the PPL authorization to express such purposes using `AuthzUseForPurpose` that allows to specify the purposes for which the data controller and processors are authorized to use the collected data. In addition, with the duration attribute for purposes, one can specify different durations for different purposes. Figure 3 shows an example of such authorization definitions.

```
<a-ppl:AuthzUseForPurpose>
<!-- Authorization for following purposes-->
  <a-ppl:Purpose duration=2Y>diagnosis</a-ppl:Purpose>
  <a-ppl:Purpose duration=5Y>research</a-ppl:Purpose>
</a-ppl:AuthzUseForPurpose>
```

Figure 3: Authorization for the specified list of purposes

Obligation 3: Breach notification In case of security or personal data breaches, cloud providers X and Y must notify M, which in turn must notify the hospital and the hospital must notify the patients. A-PPL provides a way to notify those actors using the `ActionNotify` element. Figure 4 shows an example of policy that makes the data controller responsible for notification in case of a policy violation (be it a security or a privacy breach) or a loss of data. A-PPL, via PPL, provides two such triggers.

```
<Obligation>
 <!-- Notify the data subject when triggered -->
 <TriggersSet>
   <TriggerOnPolicyViolation/>
   <TriggerOnDataLost/>
 </TriggersSet>
 <ActionNotify>
   <Media>e-mail</Media>
   <Address>data-subject@example.com</Address>
   <Recipients>Patient:Data subject</Recipients>
   <Type>Policy Violation</Type>
 </ActionNotify>
</Obligation>
```

Figure 4: Notify the data subject in case of a breach

Obligation 4: Evidence of the correct and timely deletion of personal data Cloud providers X and Y must be able to provide evidence to the platform provider M, and M must be able to provide evidence to the hospital on the correct and timely deletion of personal data. Therefore, we can use, for example, the A-PPL `ActionLog` element to tell the data processor to track the collection, processing and deletion of personal data. Combined with the A-PPL trigger, `TriggerPersonalDataDeleted`, the logged event will constitute the requested evidence. Besides, we use the action `ActionEvidenceCollection` combined with the trigger `TriggerOnEvidenceRequestReceived` to require the data processor to collect logs for the deletion as evidence of its correctness.

Obligation 5: Location of processing Cloud providers X and Y, as well as the M Platform provider have contractual obligations towards their respective customers on the location of the data processing. In order to be sure that the personal data is not shipped towards location that are not authorized, A-PPL extends XACML with the `environment:region` attribute to be placed in the XACML tag <`Environment`> tag. For example, we specify in Figure 5 that only access requested from Europe to the data targeted by this policy are permitted. If that requester cannot access the data, then it cannot move its location.

6 Related Work

Contemporaneous work by Butin et al. [11] leverages PPL to design logs for accountability. They identify the lack of expressiveness of PPL `ActionLog` which does not provide sufficient information in the logs. Besides, they discuss the fact that the PPL element `ActionNotifyDataSubject` does not allow to specify the content of the notification. Our accountability language A-PPL proposes a solution for these two above problems.

Similarly, Henze et al. [12] identify location of storage and duration of storage as the two main challenges in cloud data handling scenarios. They propose to use

```
<Rule  RuleID="write" Effect="Permit">
 <Target>
  <Environment>
   <EnvironmentMatch MatchId="string-equal">
   <AttributeValue DataType="string">
           Europe
   </AttributeValue>
   <Environment AttributeDesignator DataType="string"
     AttributeId="environment:region"/>
   </EnvironmentMatch>
  </Environment>
 </Target>
</Rule>
```

Figure 5: Control location of data in Europe

PPL to specify *data annotations* that contain the data handling obligations (e.g "delete after 30 days"). Without giving more details, they propose to extend PPL with an attribute that specifies a maximum and a minimum duration of storage and with an element that restricts the location of stored data. A-PPL also addresses these two challenges and we give in Section 4 the details of the extensions that solve these issues.

7 Conclusion

The amount of personal data published and stored within a cloud environment raises many accountability issues. We believe that machine-readable policies are a suitable means to mitigate the accountability risks akin to such a paradigm. In this paper, we consider regulations and contractual agreements from which we extract accountability obligations. From these obligations, we derive our design requirements for an accountability policy language. Having reviewed and analyzed the state of the art of policy languages, we identify the limitations of these languages and select PPL as a good candidate language. We propose then A-PPL, an extension of PPL, that handles access control, usage control and accountability specific requirements. This extension suggests new elements to specify notification, logging and evidence collection. Moreover, we describe an initial architecture for A-PPLE, the policy engine that aims at enforcing A-PPL policies. Finally, we present a concrete use case of medical sensor networks in the Cloud. From this use case, we extract several obligations and we address them by defining A-PPL policy statements.

Our future research work will consist in the finalization of A-PPLE and on its integration within a real setting that combines different tools that enforce the accountability concepts (such as an audit system).

8 Acknowledgments

This work was supported by the European Commission's Seventh framework A4Cloud[3] project. We particularly want to thank Dimitra Stefanatou for her collaboration on the analysis of the accountability obligations.

References

[1] A. Lin and N.-C. Chen, "Cloud computing as an innovation: Percepetion, attitude, and adoption," *International Journal of Information Management*, vol. 32, no. 6, pp. 533 – 540, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0268401212000539

[2] European Parliament and the Council of the European Union, "Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data ," 1995.

[3] S. Bradshaw, C. Millard, and I. Walden, "Contracts for clouds : Comparison and analysis of the terms and conditions of cloud computing services," Queen Mary, University of London, Tech. Rep. Legal Studies Research Paper 63/2010, 2010, legal Studies Research Paper 63/2010. [Online]. Available: http://ssrn.com/abstract=1662374

[4] S. Pearson, V. Tountopoulos, D. Catteddu, M. Sudholt, R. Molva, C. Reich, S. Fischer-Hubner, C. Millard, V. Lotz, M. Jaatun, R. Leenes, C. Rong, and J. Lopez, "Accountability for cloud and other future internet services," in *2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, 2012, pp. 629–632.

[5] S. Trabelsi, G. Neven, D. Raggett, C. Ardagna, C. Bournez, L. Bussard, M. Bezzi, J. Camenisch, S. de Capitani di VIMERCATI, F. Gey, A. Kuczerawy, S. Meissner, G. Neven, A. Njeh, S. Paraboschi, E. Pedrini, S. Foresti, U. Pinsdorf, F.-S. Preiss, J. Sendor, C. Tziviskou, D. Raggett, T. Roessler, P. Samarati, J. Schallaboeck, S. Short, D. Sommer, M. Verdicchio, and R. Wenning, "D5.3.4 - report on design and implementation of the primelife policy language and engine," http://primelife.ercim.eu/images/stories/deliverables, Primelife Project, Deliverable, 2011.

[6] C. A. Ardagna, L. Bussard, S. De Capitani Di Vimercati, G. Neven, S. Paraboschi, E. Pedrini, S. Preiss, D. Raggett, P. Samarati, S. Trabelsi, and M. Verdicchio, "Primelife policy language," http://www.w3.org/2009/policy-ws/papers/Trabelisi.pdf, 2009.

[3] http://www.a4cloud.eu/

[7] OASIS Standard, “eXtensible Access Control Markup Language (XACML) Version 3.0. 22 January 2013,” http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html, 2013.

[8] A. Garaga, A. S. de Oliveira, J. Sendor, M. Azraoui, K. Elkhiyaoui, R. Molva, M. Önen, R.-A. Cherrueau, R. Douence, H. Grall, J.-C. Royer, M. Sellami, M. Südholt, and K. Bernsmed, “Policy representation framework,” To be published, Cloud Accountability (A4Cloud) Project, Deliverable (To be published), 2013.

[9] HERAS AF team, “HERAS AF (Holistic Enterprise-Ready Application Security Architecture Framework),” http://herasaf.org/.

[10] K. Bernsmed, H. Kuan, and C. Millard, “Deploying Medical Sensor Networks in the Cloud - Accountability Obligations from a European Perspective,” *Submitted for publication*, 2014.

[11] D. Butin, M. Chicote, and D. Le Métayer, “Log design for accountability,” in *Security and Privacy Workshops (SPW), 2013 IEEE*. IEEE, 2013, pp. 1–7.

[12] M. Henze, M. Großfengels, M. Koprowski, and K. Wehrle, “Towards data handling requirements-aware cloud computing,” in *2013 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2013.

[13] M. Marchiori, “The platform for privacy preferences 1.0 (P3P1.0) specification,” W3C, W3C Recommendation, Apr. 2002, http://www.w3.org/TR/2002/REC-P3P-20020416/.

[14] M. Y. Becker, A. Malkis, and L. Bussard, “S4P: A generic language for specifying privacy preferences and policies,” *Microsoft Research*, 2010.

[15] I. Aktug and K. Naliuka, “ConSpec – a formal language for policy specification,” in *Electronic Notes in Theoretical Computer Science*. Elsevier, 2008, vol. 197, pp. 45–58. [Online]. Available: http://ac.els-cdn.com/S1571066108000480/1-s2.0-S1571066108000480-main.pdf?_tid=096c063e-2371-11e2-814c-00000aacb360&acdnat=1351698097_1121c29a6754c6d17db46c653fb85c6b

[16] N. Damianou, N. Dulay, E. Lupu, and M. Sloman, “The ponder policy specification language,” in *POLICY*, 2001, pp. 18–38.

[17] A. Barros and D. Oberle, *Handbook of Service Description: USDL and Its Methods*. Springer Publishing Company, Incorporated, 2012.

[18] D. D. Lamanna, J. Skene, and W. Emmerich, “SLAng: A Language for Defining Service Level Agreements,” in *Proceedings of the The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 100–. [Online]. Available: http://dl.acm.org/citation.cfm?id=795675.797134

[19] OASIS Web Service Security (WSS) TC, "Web Services Security: SOAP Message Security 1.1," https://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf, 2006.

[20] OASIS Web Services Secure Exchange (WS-SX) TC, "WS-Trust 1.4," http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/errata01/os/ws-trust-1.4-errata01-os-complete.html, 2012.

[21] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible markup language (xml)," *World Wide Web Journal*, vol. 2, no. 4, pp. 27–66, 1997.

A Survey of Policy Languages

We present here the results of our analysis of existing policy languages. A detailed version of the survey can be found in [8]. The first step of our analysis is to check whether the languages satisfy the requirements. None of the languages we reviewed meets all the policy language requirements. However, they may all fulfill a subset of these requirements. For example, XACML covers (or partially covers) requirements R1, R2, and R4 whereas PPL covers (or partially covers) all the requirements except R6 and R7. From this first analysis, we classify our selected policy languages into four categories: (i) *Access Control:* eXtensible Access Control Markup Language (XACML, [7]); (ii) *Privacy:* The Platform for Privacy Preferences (P3P, [13]), the Primelife Policy Language (PPL, [6]) and SecPal for Privacy (SecPal4P, [14]); (iii) *Policy specification for security:* Conspec ([15]) and Ponder ([16]); (iv) *Service Description:* The Unified Service Description Language (USDL, [17]), SLAng ([18]) and WS-Policy ([19, 20]). Note that these categories are not exclusive, meaning that one language can fit into several categories. For instance, PPL allows to specify access control rules as XACML. We also argue that we cannot define from our set of languages an additional category *Accountability language*. In particular, most of the languages do not provide means to express logging, reporting and audit obligations. Therefore, the design of the accountability language we propose in the following sections represents an unprecedented attempt to express accountability obligations via a policy language.

In a second step, we study the extensibility of the reviewed languages in order to extend one of the languages with accountability features. We focus on XML-based languages, since XML (the eXtensible Markup Language [21]) provides many extension points to extend the syntax and the vocabulary of the language. In addition, XML is a standard and well documented. Thus adding extension to an XML-based language is fairly simple. Languages such as XACML, P3P and PPL use XML to define policies related to access control and privacy. So we give priority to these languages for our work.

As a result of this survey, we focus our effort on the extension of PPL.

B List of available PPL Triggers and Actions

We depict here the list of available triggers and actions in the PrimeLife Policy Language. A trigger is an event filtered by a condition that generate actions. Triggers and actions are part of PPL obligation model. A-PPL extends this list with new or enhanced triggers and actions to make it compliant with the design requirements of an accountable policy language.

Table 5: List of Triggers and Actions in PPL

Name	**Description**
Triggers	
TriggerAtTime	Occurs based on a particular defined time
TriggerPeriodic	Occurs repeatedly according to a well-specified period
TriggerPersonalData-AccessedForPurpose	Occurs each time the personal data bound to the obligation is accessed of one of the defined purposes
TriggerPersonalDataDeleted	Occurs when the personal data associated with the obligation is deleted
TriggerPersonalDataSent	Occurs when the personal data akin to the obligation is forwarded to a third-party
TriggerDataSubjectAccess	Occurs when the data subject requests access to ts own personal data collected by the data controller
Actions	
ActionDeletePersonalData	Deletes a piece of personal data (data retention)
ActionAnonymizePersonalData	Anonymizes a particular piece of data
ActionNotifyDataSubject	Notifies the data subject when triggered, that is, send the information concerning the event that triggers the obligation to the data subject
ActionLog	Logs an event, that is, writes in a log file the information concerning the event that triggers the obligation
ActionSecureLog	Logs an event and ensures integrity and authentication of origin of the event