An Access Control Language for a General Provenance Model

Qun Ni[1], Shouhuai Xu[2], Elisa Bertino[1], Ravi Sandhu[3], and Weili Han[4]

[1] Purdue University, Department of Computer Science, West Lafayette IN, USA
{ni,bertino}@cs.purdue.edu
[2] UT San Antonio, Department of Computer Science, San Antonio TX, USA
shxu@cs.utsa.edu
[3] UT San Antonio, Institute for Cyber Security, San Antonio TX, USA
ravi.sanhdu@utsa.edu
[4] Fudan University, Software School, Shanghai, China
wlhan@fudan.edu.cn

Abstract. Provenance access control has been recognized as one of the most important components in an enterprise-level provenance system. However, it has only received little attention in the context of data security research. One important challenge in provenance access control is the lack of an access control language that supports its specific requirements, e.g., the support of both fine-grained policies and personal preferences, and decision aggregation from different applicable policies. In this paper, we propose an access control language tailored to these requirements.

1 Introduction

Provenance, a documented history of an object, has already been widely used in the scientific and grid computing domains to properly document workflows, data generation, and processing. Access control of provenance is of the highest importance for many critical organizations [1] either because the fulfillment of their duties relies on a secure provenance management or because the protection of provenance is required by laws or regulations. In a national security agency, the improper disclosure of the source or the ownership of a piece of classified information may result in great and irreversible losses [2]. In a pharmaceutical company, the source of data and the processing executed on data may be sensitive or valuable. In the absence of an access control mechanism for protecting such information, malicious or faulty insiders could steal it [1]. Additionally, many compliance regulations require *proper* archives and audit logs for electronic records [1], e.g. HIPAA mandates to properly log accesses and updates to the histories of medical records.

Therefore, provenance access control is considered to be the primary issue in provenance security [3]. Unfortunately, despite the large number of research efforts focusing on the management of provenance [4, 5, 6, 7, 8], only a few of these efforts have investigated the problem of securing provenance [3, 2, 9, 10, 1]. Moreover, none of these proposals focuses on access control.

W. Jonker and M. Petković (Eds.): SDM 2009, LNCS 5776, pp. 68–88, 2009.

The problem of access control for provenance is complicated by the fact that given a request to access some provenance information, different access control policies, possibly from different sources, may apply (see Figure 1): organizational high-level security policies, departmental fine-grained access control policies, privacy laws and regulations. Moreover, individuals who contributed to the information, referred to as originators, may specify personal preferences on the disclosure of such information. Given an access request, whether the request is allowed or not depends on the decisions from all of these policies. We thus need a language able to support the specification of fine-grained policies, privacy policies, and preferences, and equipped with a flexible access control decision aggregation mechanism.

The goal of this paper is to propose such a comprehensive access control language addressing those specific requirements of provenance access control, e.g. fine-grained, privacy-aware, and originator control. Our contributions include:

- A novel provenance model that captures the characteristics of previously proposed provenance models and is the base for analyzing the requirements for provenance access control.
- A language tailored to fine-grained provenance access control and originator preferences.
- A simple yet flexible evaluation mechanism for decision aggregation.

The rest of this paper is organized as follows: Section 2 introduces our general provenance model and analyzes the requirements of provenance access control. Based on such a provenance model, Section 3 develops the access control language model. Section 4 discusses the flow of access control decision process. Section 5 shows how originator preferences are taken into account in access control decisions. Section 6 illustrates our approach with several examples. Section 7 discusses related work. Section 8 outlines some conclusions and directions for future work.

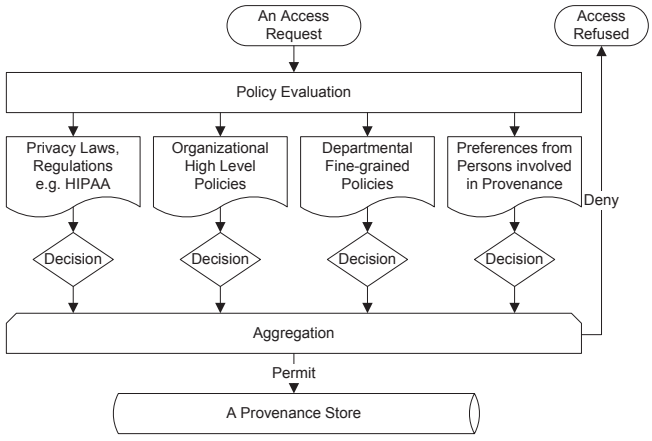

Fig. 1. Different policies may be applicable

2 A Provenance Model

In order to develop an access control language for provenance, the first step is to analyze the requirements for a provenance access control model. Our analysis is based on the sensitivity of the different entities in a provenance model that describes how provenance is represented.

2.1 The Model

Unfortunately, there is currently no standard for representing provenance in spite of some initial attempts such as the Open Provenance Model [11] and the Architecture for Provenance Systems [3]. Some proposals [4, 5, 6, 7] focus on different application domains (scientific data provenance vs electronic health record), have different forms (relational vs XML), or purposes (storage vs query). Several systems for managing data lineage and provenance are being used in the context of scientific processes, e.g. Chimera [12], myGRID [8], and ESSW [13]. Moreover, some workflow systems [14] are also able to generate provenance information as well.

Provenance is already well understood in the field of art history where it refers to the trusted, documented history of some art objects [3]. Given a documented history, the object attains an authority that allows scholars to understand and appreciate its importance and context relative to other objects. Art objects that do not have a trusted, proven history may be treated as faked items. This same provenance concept may also be applied to data acquired, generated, manipulated, and distributed by computer systems and applications. One of our primary objectives is thus to define a provenance representation that is suitable for such data. Hence, in this context, we give the following definition of the provenance of data (see Fig. 2). Our provenance model can capture and describe provenance models proposed by aforementioned approaches and research.

Definition 1 (Provenance). *The provenance of a piece of data is the documentation of messages, operations, actors, preferences, and context that led to that piece of data.*

An operation is a manipulation performed on or caused by some data, referred to as input messages, and resulting in other data, referred to as output messages. Messages represent data flows between operations. Applications, database

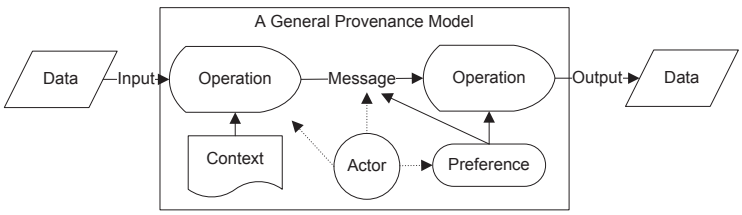

Fig. 2. A provenance model

commands, and web services are typical examples of operations, while copy and paste, emails, and inter-communication between UNIX processes are typical messages. The piece of data with which the provenance is associated is the output of the last operation in the provenance.

Operations and messages are operated by actors that could be application logics, workflow templates, or human beings. In some situations, information about actors, e.g., a physical therapist of a treatment of musculoskeletal disorders in a patient, is also a necessary component of provenance. Such an observation motivates the introduction of actor records in our provenance model.

Context refers to additional data which is independent of the input messages of an operation but affects the content of the output messages of the operation, e.g., operation states and operation parameters. Some operations are stateful or rely on values from some external context variables. In some circumstances, the internal states of a stateful operation and the values of external variables may also be necessary in order to understand the functionality or performance of the operation and therefore the nature of the result of the operation [3]. Moreover, in scientific computations, the parameters used in some operations are crucial, like for example the parameters in a classification algorithm, for the final output [6]; thus such information should also be included in the provenance as context records.

Most existing provenance studies do not consider security and privacy requirements concerning the utilization of provenance, especially the requirements concerning actors. Security and privacy are however crucial when provenance contains information of commercial value or of a legally sensitive nature (e.g., a proprietary algorithm). Usually such sensitive information is very specific, and its protection requirements may depend on the specific application domain and can often only be determined by the involved actors. Thus there is a need for a provenance model able to address such requirements in order to limit the access to the operation or message content based on access restriction by the corresponding actor [9]. Such an observation motivates preference records. These records are designed for actors to specify their personal preferences that control whether and how other actors may utilize operation and message records.

If we consider connections resulting from actors, the context, and preferences to be special messages, these records generally form a directed acyclic graph (DAG) with messages as edges and other records as nodes. Such graphs may have cycles when representing provenance for workflow; however, we can always rewrite a graph with cycles into a DAG by replicating edges and nodes [7].

2.2 Provenance Records

Provenance is represented by a set of *provenance records* stored in a *provenance store*. Such a store can be implemented by various systems, like relational DBMS or XML document management systems. Based on the proposed provenance data model (see Fig. 2), we have defined five kinds of provenance records, that is: operation records, message records, actor records, preference records, and context records. To be general, we leave out unspecified details about the

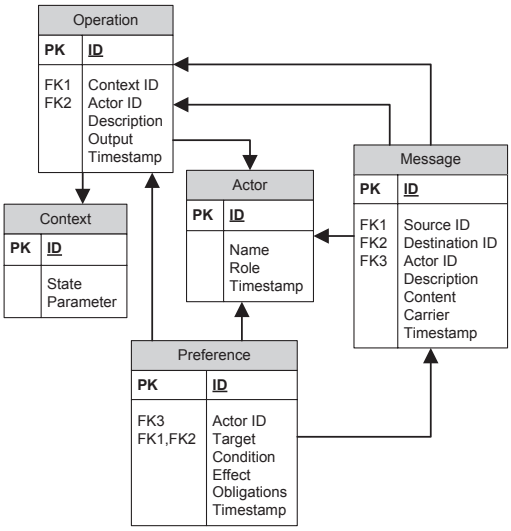

Fig. 3. Provenance record schemata

implementation of these records. However, to illustrate the usage of provenance records and the access control requirements of a provenance store, it is necessary to consider some of those details. In what follows, we discuss details about each type of record that are relevant to the definition of provenance access control. The schema of each record is shown in Fig. 3; in the graphical representation, PK means "primary key" and FK means "foreign key".

Each record consists of several attributes. Some attributes are optional in that their value might be null. A basic assumption in the provenance model is that each piece of data and each provenance record are uniquely identified by one identification attribute, referred to as the ID attribute. Message, operation, actor, and preference records have a timestamp attribute which is useful for time-restricted provenance queries and preference evaluation (their use will be discussed in Section 5).

Operation record attributes include ID, actor ID, context ID, description, output, and timestamp. The detail of a description attribute depends on applications. The description attribute may clearly define a function by pseudo-code, or even by source code, but it can also be only a function name. The output attribute describes the output of the operation. The value of the output attribute usually represents the connection between provenance records and data records.

Message record attributes include ID, actor ID, source ID, destination ID, description, content, carrier, and timestamp. Specific details about the description attribute depend on applications. A message record is not a copy of a *real* message between two operations; it is just the provenance of the real message. For the purpose of provenance completeness, the message content attribute will be expected to contain the full information transmitted by the real message. However, other choices are possible. If intermediate data transferred in the real

message have been stored elsewhere, the reference ID of the intermediate data can be stored in the content attribute instead those in the data. Moreover, if the destination operation is reversible and such intermediate data may be reproduced, the data need not be stored in the content attribute either. The carrier attribute indicates the message transferring channel, e.g. email, which may be sensitive and useful in some cases, e.g. digital forensics [1].

Actor record attributes include ID, name, and role. Actors usually have names and roles. A role, like the concept of role in role-based access control, is a job function of the actor. Someone may argue why not use the actor information directly from human resource databases. A human being may have different roles during his/her career time. Thus he/she may have different versions of actor records with different roles for different operation / message / preference records. This is the crucial reason why we cannot rely only on the information from the actor records stored in human resource databases. Such a record usually only stores the latest actor information, but an actor record in a provenance store needs to record complete historical actor information.

Context record attributes include ID, state, and parameter. The content of a context record heavily depends on the application domain. Usually each operation record has at most one context record. It seems preferable to include the context record into the operation record. We choose to separate the context record from the operation record because of two reasons. First, the schema and size of context records vary with respect to different operation records. Some operation records do not have a context record; however other operation records may have a complex context record. Second, it is also possible that two different operation records may share the same context record. Context records are the only provenance records that do not need timestamps because their timestamps are determined from their parent operation records.

Preference record attributes include ID, actor ID, target, condition, effect, obligations, and timestamps. Preference is used to record the access preferences of the actor of the operation or message. Sometimes it is also useful to record the preferences expressed by the subject of the operation/message, for example a patient in the case of healthcare applications. The actor ID attribute is used to record the author of the preference record. Authors are usually actors. A patient may specify his/her preference, but the preference is usually recorded by a nurse or a doctor. The target attribute is used to specify the subject and the exact record at which the preference aims. Each target of a preference record only references an operation record or a message record. Details of the target and other attributes, e.g. conditions, are elaborated in Section 3.

Because provenance is a documented history of a piece of data, it has been pointed out [10] that a provenance store is immutable. However, it is reasonable to allow some actors to rationally change their preferences on their own records. There are two approaches to support such selective updates. One is to allow those changes to overwrite previous values. The other approach is to use versioning and associating with each preference record a timestamp. Given a query, if two

preferences from a same actor evaluate to one permit and one deny, the result from the latest preference record takes precedence. We adopt the latter solution because

- previous preference records are a part of data provenance and have a value as well;
- the immutable property reduces the complexity of access control on provenance because we only need to focus on querying and do no need to worry about changes, such as updates and deletions, to existing provenance records. Such a property also makes it possible to store provenance in "Write Once, Read Many" (WORM) devices that may greatly help in protecting provenance integrity.

As shown in Fig. 3, there are relations between the records that compose a provenance DAG. The actor ID in an operation, message or preference record references the primary key of an actor record. The source ID and destination ID in a message record reference two primary keys in operation records. The context ID in an operation record references the primary key of a context record. The record field together with the restriction field (Section 3) in the target of a preference record references the primary key of either an operation record or a reference record.

Another basic assumption in our provenance model is that, at each time instant, a piece of data is at most manipulated by *one* operation. The piece of data can be manipulated several times by the same or different operations, and such an operation history builds the provenance of the piece of data.

2.3 Provenance Records for Medical Data

We now discuss the application of the proposed provenance model to represent the provenance of medical records generated from the Diabetes Quality Improvement Program workflow shown in Fig. 4, where CDC refers to Comprehensive Diabetes Care. Medical records and relevant provenance records generated from the workflow in Fig. 4 are shown in Fig. 5. The first column shows medical records, e.g. register, eye exam etc., except for actors. Other provenance records are shown on the right side. For simplicity, we do not show some attributes, e.g. the timestamps in actor records, and some records, e.g. context records.

Based on the records reported in Fig. 5, we have the following observations:

- Each medical record is generated by one operation at a specific time, and can be uniquely identified by the output attribute (with two fields) in the operation's record.
- Some message records have values in their content attributes that reference medical records, and others do not.
- Message records and operation records connected by these message records form two independent DAGs whose structure is exactly the same as that of the workflow of interest (Fig. 4).

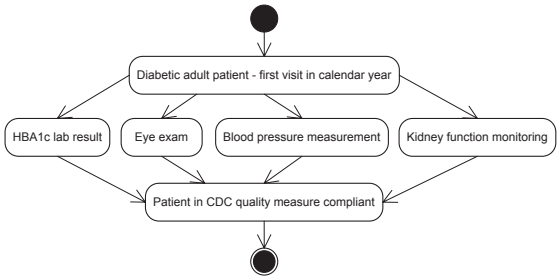

Fig. 4. Diabetes QI Program workflow

Medical records

Register

ID	Name
1	Alice
2	Bob

Eye_exam

ID	Patient ID	Retinopathy
3	1	Yes
4	2	No

HBA1c

ID	Patient ID	Result
7	1	6.50%
8	2	8.30%

Blood_Pressure

ID	Patient ID	Result
2	1	125-85
3	2	144-95

Kidney_Function

ID	Patient ID	Compliant
5	1	Yes
6	2	No

CDC

ID	Patient ID	Status
8	1	Good
9	2	Bad

Provenance records

Operation

ID	Actor ID	Context ID	Description	Output.record	Output.id	Timestamp
1	1	null	registration	Register	1	1/23/2009 6:00
2	1	null	registration	Register	2	1/24/2009 6:14
3	2	null	eye examination	Eye_exam	3	1/25/2009 6:28
4	2	null	eye examination	Eye_exam	4	1/26/2009 6:43
5	5	null	HBA1c test	HBA1c	7	1/27/2009 6:57
6	5	null	HBA1c test	HBA1c	8	1/28/2009 7:12
7	4	null	Blood pressure	Blood_pressure	2	1/29/2009 7:26
8	4	null	Blood pressure	Blood_pressure	3	1/30/2009 7:40
9	3	null	Kidney function	Kidney_Function	5	1/31/2009 7:55
10	3	null	Kidney function	Kidney_Function	6	2/1/2009 8:09
11	6	null	CDC	CDC	8	2/2/2009 8:24
12	6	null	CDC	CDC	9	2/3/2009 8:38

Actor

ID	Name	Role
1	Jame	Nurse
2	Katty	Practitioner
3	John	Doctor
4	David	Nurse
5	Tom	Practitioner
6	Betty	Doctor

Message

ID	Actor ID	Carrier	Description	Content.record	Content.id	Timestamp	Src ID	Des ID
1	1	paper	Eye exam req	null	null	1/23/2009 8:24	1	3
2	1	paper	Eye exam req	null	null	1/24/2009 8:52	2	4
3	1	paper	HBA1c test req	null	null	1/25/2009 9:21	1	5
4	1	paper	HBA1c test req	null	null	1/26/2009 9:50	2	6
5	1	paper	Blood pressure req	null	null	1/27/2009 10:19	1	7
6	1	paper	Blood pressure req	null	null	1/28/2009 10:48	2	8
7	1	paper	Kidney function req	null	null	1/29/2009 11:16	1	9
8	1	paper	Kidney function req	null	null	1/30/2009 11:45	2	10
9	2	email	Eye exam result	Eye_exam	3	1/31/2009 12:14	3	11
10	5	email	HBA1c test result	HBA1c	7	2/1/2009 12:43	5	11
11	4	email	Blood pressure	Blood_Pressure	2	2/2/2009 13:12	7	11
12	2	email	Eye exam result	Eye_exam	4	2/3/2009 13:40	4	12
13	5	email	HBA1c test result	HBA1c	8	2/4/2009 14:09	6	12
14	4	email	Blood pressure	Blood_Pressure	3	2/5/2009 14:38	8	12
15	3	email	Kidney function	Kidney_Function	6	2/6/2009 15:07	10	12
16	3	email	Kidney function	Kidney_Function	5	2/7/2009 15:36	9	11

Preference

ID	Actor ID	Target. Subject	Target. Record	Target.Restriction	Condition	Timestamp	Effect	Obligs
1	3	actor	operation	actor.role = doctor and operation.id = 10	purpose = research	1/23/2009 6:00	necessary permit	null
2	5	actor	operation.body	operation.id = 5 and actor.name = David	null	1/27/2009 6:57	deny	null
3	3	actor	message.body	message.id = 16	purpose= marketing	2/7/2009 15:36	deny	null

Fig. 5. Medical records and Provenance records

– Actor records are referenced from operation, message, and preference records.
– Each preference record references exact one message record or operation record.

All records other than preference records are easily understood. The meaning of preference records will be more clear in Section 5. One important design choice in our provenance model is that we do not need a provenance pointer to be included in the original data item to indicate the location of relevant provenance records. Given an item ID, we can directly retrieve its provenance from the provenance store based on our model. An advantage of our model is that it does not need the adjustment of the schemata of existing datasets. It is well known that database administrators usually "hate" such adjustments.

2.4 Desiderata for a Provenance Access Control Model

Based on previous work on securing provenance [10, 3, 1] and query examples in provenance management [6, 5, 15], we identify some important requirements of an access control mechanism for provenance that are discussed in what follows.

First, provenance access control must be fine-grained. Because of the sensitivity of different provenance records, it is usually the case that an organization may want to ensure that certain portions of the provenance records be only accessible to certain parties, e.g. a few treatments in privacy sensitive electronic healthcare records [9], sources of information in a classified document by the Central Intelligence Agency [2], or a proprietary algorithm applied to some segments of scientific data [1]. Such a requirement asks for the ability to confine a query to a very limited scope with respect to subjects and/or objects in terms of access control. Moreover, it may also be useful to ensure that certain subjects are authorized to access only the subset of the provenance records that are necessary for a specific purpose or more generally, any type of context[1] in which provenance representations can be useful [3]. This would require the ability to express authorizations with context restrictions.

Second, provenance access control may have to constrain data accesses in order to address both security and privacy. One typical example is in the context of the electronic heathcare records that essentially contain both original data and their provenance. If we consider the final medical results about the treatment of a patient to be a piece of data, its provenance usually contains observations, procedures, tests, prescriptions, and information flows between patients, doctors, practitioners, and nurses. Therefore accesses to electronic heathcare records should not only comply with organizational security policies based on well-known principles such as "need to know" and "least privilege", but also comply with privacy regulations, such as HIPAA.

Third, provenance access control may need both originator control [16, 17] (ORGCON) and usage control [18, 19] (UCON). ORGCON is an access control proposal that requires recipients to gain originator's approval for a re-dissemination of an originally disseminated digital object or a new digital object that includes the originally distributed digital objects. Motivated by digital rights management, UCON is an access control model that confines the usage of re-disseminated digital objects. As mentioned in Section 2.1, a provenance access control must be able to take into account preferences by actors about how to utilize relevant records, which is indeed similar to an originator (actor) control in usage control (record usage). One challenge from such requirement is that provenance access control should provide a meaningful and usable method to integrate decisions from both organizational policies and actor preferences, for which multiple versions may exist. Another challenge is the need of a mechanism that ensures regulations, e.g. HIPAA, always take precedence over preferences when there is a conflict.

[1] Here the meaning of the term "context" is different from that of the term "context" in which an actor performs some operation.

3 An Access Control Language for Provenance Stores

In this section we propose an access control language, based on our provenance model, for addressing the requirements discussed in the previous section. The language supports the specification of both actor preferences and organizational access control policies.

3.1 The Language

The proposed language is graphically represented in Fig. 6. Its main components are *target*, *condition*, *effect*, and *obligations*, which are discussed in what follows.

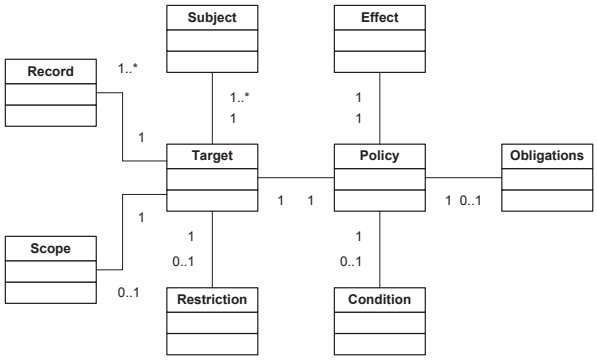

Fig. 6. An Access Control Language

3.2 Target

Since a provenance store is immutable, only two operations can be supported: *append* and *read*. We believe that in a provenance-aware system, the *append* operation should be automatically performed by applications and not by users, like *log* operations in database systems. The privilege to stop or start an *append* operation by an application is not controlled by regular users but by administrators. Therefore, our access control language only focuses on query (read) operation on provenance records.

The target specifies the set of *subjects* and *records*, to which the policy is intended to apply. Because the provenance store is immutable and the access on which we focus is query (read), the type of access, e.g. read or append, is intentionally omitted in the target. The subject element can be the name of any collection of users, e.g. actor or professor, or a special user collection *anyuser* which represents all users. The record element can be the name of any collection of provenance records, e.g. operation, some attributes in records, e.g. operation.body, or a special record collection *anyrecord*. The following example shows a policy target that applies to access requests from any user and to all information contained in the attribute description in the operation records.

```
<target>
    <subject>anyuser</subject>
    <record>operation.description</record>
</target>
```

The (optional) *restriction* element can further refine the applicability established by the target through the specification of predicates on subject attributes (combined with *anyuser*) and/or record attributes (Section 2.4). The following target example applies to users with a *doctor* role and to the description of operation records before year *2009*.

```
<target>
    <subject>anyuser</subject>
    <record>operation.description</record>
    <restriction>anyuser.role == doctor AND operation.timestamp <=1.1.2009</restriction>
</target>
```

Provenance represents the lineage of a piece of data; thus when we define a provenance access control policy we often want the ability to specify one policy for all provenance records related to a piece of data. Such a requirement is captured by an optional element *scope*. Two predefined values, "transferable" and "non-transferable" (default) can be specified in the element, where "transferable" means that the target not only contains the set of records defined by other elements in a target, but also includes all ancestors of these records. By contrast, "non-transferable" means that the target only contains the set of records defined by other elements. If the *scope* element is absent from a target, the "non-transferable" semantics is adopted for the target. The target of the following example shows a provenance record set for an operation record and its antecedent records that resulted in the CDC record of patient Alice (See Fig. 5).

```
<target>
  <subject>anyuser</subject>
  <record>operation</record>
  <restriction>operation.id == 11</restriction> <!-operation 11 generates CDC status->
  <scope>transferable</scope>
</target>
```

3.3 Condition

A condition represents a boolean expression that describes the optional context requirements (Section 2.4) that confine the applicable access requests, e.g. access purpose, limitation on access time and location, and verification of the record originator's license. System or context variables usually appear in the condition expression. The following condition restricts an applicable access to be executed only from a machine, e.g. *obelix*, and with an access purpose, e.g. *research*.

```
<condition>system.machineid == obelix AND purpose == research</condition>
```

Restrictions and conditions are both boolean expressions, and are crucial in order to achieve fine-grained access control. The reason why they are mapped onto different components is not only because they focus on different policy aspects, i.e. the target scope and the context requirements, but also because they have a different impact on the aggregation of decisions by different applicable policies. We will elaborate more on this issue in Section 4.

3.4 Effect

The effect of a policy indicates the policy author's intended consequence of a "true" evaluation for policy. In the current version of the language, the effect can take one of the following values: *Absolute Permit*, *Deny*, *Necessary Permit*, and *Finalizing Permit*. The motivation and semantics of these four different effects will be discussed in Section 4.

The following example shows a policy without obligations. The policy requires that any doctor who accesses the description field of operation records before year *2009* can only do so from machine *obelix* and that the access purpose must be *research* only.

```
<policy ID=1>
  <target>
    <subject>anyuser</subject>
    <record>operation.description</record>
    <restriction>anyuser.role == doctor AND operation.timestamp <1.1.2009</restriction>
  </target>
  <condition>system.machineid == obelix AND purpose == research</condition>
  <effect>necessary permit</effect>
</policy>
```

3.5 Obligations

An obligation is an operation, specified in a policy, that should be executed before the condition in the policy is evaluated, in conjunction with the enforcement of an authorization decision, or after the execution of the access. There are at least two cases in which we may need obligations in an access control language for provenance stores. An actor may require any user of his/her records to obtain his/her agreement before access to these records, or to inform him/her after the access. He/she may do so by adding a pre-obligation or a post-obligation [20] in his/her preference. Another case is when one has to comply with regulations that include obligations, e.g. HIPAA; organizational policies may also require obligations. Due to space limitations, we do not elaborate on the obligations component and we only provide an example. We refer the interested readers to our previous work [20]. The following example shows an obligation specifying that the actor of the record has to be informed about each data access within 10 days from the date of the access. Commonly used obligations are "inform actors or originators of an access" and "obtain (either advance or later) approval from actors or originators". Operations in an obligation may have their specific parameters and we adopt a simplified representation of operations.

```
<obligations>
  <obligation>
    <operation>inform the actor of the record</operation>
    <temporal constraint>10 days</temporal constraint>
    <fulfill on>access</fulfill on>
  </obligation>
</obligations>
```

4 Policy Evaluation

Abstractly, we may consider a two-dimensional space defined by a provenance store query, referred to as a *query space*, to be a tuple of a singleton of user and a set of records, represented by the pair (user_q, records_q). If we consider the tuple of the sets of subjects and records defined by a target to be a *target space*, represented by (subjects_t, records_t), then we have the following definition.

Definition 2 (Applicable Policy). *A policy is applicable to a query if and only if the component-wise intersection of the target space of the policy and the query space generates neither an empty user set nor an empty records set.*

In other words, a policy is applicable to a query if and only if its target space contains both the query user and a subset of query records. Given a query, only the conditions, effects, and obligations in applicable policies are evaluated in its final authorization decision. The *evaluation* of an applicable policy is its *condition evaluation result*, either *true* or *false*. The evaluation sequence depends on the effects of applicable policies, as shown in Fig. 7. The final decision depends on both the effects and the condition evaluation results of applicable policies.

An applicable policy with an absolute permit effect has the highest priority. Given a query, if at least one applicable absolute policy evaluates to true, the query is permitted regardless of the effects of other applicable policies. The motivation for the absolute permit is that provenance queries required by law enforcement institutions or national security agencies should be able to circumvent the limitation specified by actor preferences and organizational policies.

A policy with a deny effect has the second priority. Given a query, if no applicable absolute permit policies evaluate to true and at least one applicable deny policy evaluates to true, the query is denied regardless of the effects of other applicable policies. The motivation is that negative policies may significantly reduce the total number of policies required in practice, which may in turn reduce the administration costs for policies. The popular "deny takes precedence" principle is adopted in the language.

A policy with a necessary permit effect has the third priority. Given a query, if no applicable absolute permit policies evaluate to true, no applicable deny policies evaluate to true, and at least one applicable necessary permit policy evaluates to false, the query is denied regardless of the effects of any other applicable policies. The motivation for the necessary permit is the requirements from actor preferences and regulation compliance in organizational policies. When an actor specifies his/her preferences on some operation or message records,

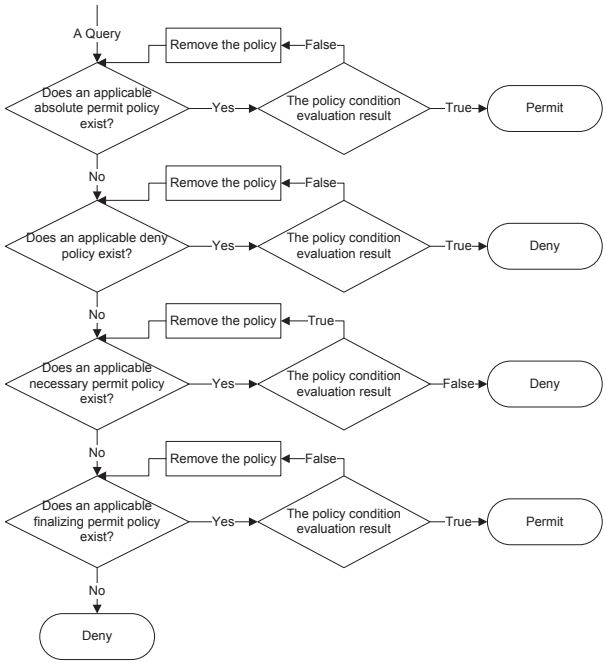

Fig. 7. Policy Evaluation Flow

he/she, like donors who regulate the usage of their funds, usually only specifies some *necessary* conditions that should be satisfied by future usage of relevant records. To comply with regulations, e.g. HIPAA, an organization has to specify corresponding access control policies that are usually not sufficiently fine-grained but necessary for all relevant queries. A necessary permit is useful in these cases.

It should be noted that we can write a deny policy which is semantically equivalent to another necessary permit policy by negating the condition and changing the effect; thus for any query, the final decisions are the same. However, since some regulations are more naturally expressed by a deny policy and other preferences are more naturally expressed by a necessary permit, we intentionally do not merge these two effects into one.

A policy with a finalizing permit effect has the lowest priority. Given a query, if no applicable absolute permit policies evaluate to true, no applicable deny policies evaluate to true, no applicable necessary policies evaluate to false, and at least one applicable finalizing permit policy evaluate to true, then the query is permitted. Otherwise, the query is denied.

One goal in classifying positive authorization policies into necessary permit policies and finalizing permit policies is to achieve flexibility and convenience for the administration of policies at different granularity levels. For instance, the Chief Security and Privacy Officer (CSPO) of an organization may specify a binding set of basic regulations for all departments with respect to the access to provenance information within the organization. These regulations can in turn be

refined by the various departments [21]. One way to address this requirement is to let the CSPO define some applicable necessary permit policies with which all the departments have to comply. However, these necessary permit policies cannot authorize access requests because they are not sufficiently fine-grained. Each department can then define its own fine-grained finalizing policies. The decision for a query is then obtained by composing the decisions from all applicable policies with at least one decision from the finalizing permit policies in one department. In other words, if all applicable necessary permit policies allow the access request and at least one finalizing policy allows the request, the query is authorized. Positive norms and negative norms are based on a similar idea [22].

5 Originator Preferences

Our access control language can be applied to specify originator preferences, that is, to support originator control. Compared to organizational policies, originator preferences are usually very specific to a particular record and its fields. The following originator preference specifies that the description information of operation record *12345678* cannot be accessed for either *reverse engineering* or *reselling* purpose. Fig. 5 shows other preference record examples.

```
<preference ID=1>
  <target>
    <subject>anyuser</subject>
    <record>operation.description</record>
    <restriction>operation.ID == 12345678</restriction>
  </target>
  <condition> purpose == reverse engineering OR purpose == reselling</condition>
  <effect>deny</effect>
  <timestamp>1.29.2009</timestamp>
</preference>
```

The timestamp plays a key role in the evaluation of the originator preferences. When multiple preferences exist, the evaluation criterion is that only the latest *applicable* preference is evaluated. The final authorization decision depends on the latest applicable preference and all applicable organizational policies.

Given a query, the applicable organizational policies and applicable preferences are evaluated together. The semantics of different effects in user preferences is the same as that of policies. Given a record to be queried, if only necessary permit preferences are specified on this record and there are no corresponding organizational finalizing permit policies, according to the evaluation flow introduced in Section 4 the record cannot be disclosed. This behavior is reasonable and meets our expectation.

6 Additional Examples

In this section, we show how our access control language can specify access control policies from some recently published papers and thus meet access control requirements identified there.

As mentioned in Section 2.4, electronic health records are a hybrid representation of data and relevant provenance, and thus need to be protected to assure privacy. Three crucial components in privacy regulations are the access or usage purpose, obligations, and conditions [23]. The proposed access control language directly supports conditions and obligations. Purpose requirements can be specified as predicates in conditions (as a matter of fact, we have already used them in previous examples). Someone may argue that the language cannot prevent policy authors from specifying invalid purposes. We, however, believe that specifying valid purposes in conditions is the duty of the policy authors and thus it is reasonable that the language itself is only capable of specifying purposes in policies. Perhaps by policy analysis we may be able to identify invalid purposes in policies that is left for our future work.

In conjunction with effects, purpose predicates can directly model the following common cases of purpose requirements in privacy regulations.

- case 1: some records can only be used for some specific purposes;
- case 2: some records can be used for some specific purposes;
- case 3: some records should not be used for some purposes.

These cases can be represented by the following three policy fragments.

```
<policy ID=1> <!-- case 1 -->
  ...
  <condition>purpose == research OR purpose == development</condition>
  <effect>necessary permit</effect>
  ...
</policy>
<policy ID=2> <!-- case 2 -->
  ...
  <condition>purpose == research OR purpose == development </condition>
  <effect>finalizing permit</effect>
  ...
</policy>
<policy ID=3> <!-- case 3 -->
  ...
  <condition>purpose == marketing</condition>
  <effect>deny</effect>
  ...
</policy>
```

The evaluation flow introduced in Section 4 can be directly applied to the integration of decisions from privacy policies.

An employee's performance review [10] is an example where the provenance is more sensitive than the data. Generally employees are permitted - and usually encouraged - to read their performance review. However, the employee is not told who had input in writing the review. Thus the employee can see the data but not the provenance of that data. The following policy forbids any subject to access the message that leads to the review document about him/her; thus no source can be disclosed.

```
<policy ID=1>
  <target>
    <subject>anyuser</subject>
    <record>operation</record>
    <restriction>operation.output.record == review AND
        anyuser.name == review.objectname</restriction>
  </target>
  <effect>deny</effect>
</policy>
```

Braun et al. [10] argued that provenance is poorly served by traditional data security models because these models focus on individual (provenance) data items, whereas provenance focuses on the relationships between those items. These relationships and data items form a DAG, and both nodes and edges need to be protected. In addition, Braun et al. also suggested that one may need to hide the participation of an operation. We agree with these requirements but do not agree that traditional data security models cannot secure provenance. With an appropriate provenance model, like the one proposed in this paper, the relationships between data items (message records) and the participation of an operation (actor records) may be secured by traditional style access control policies, as shown by examples in this paper.

As indicated by Hasan [1], the ownership history of documents (e.g., the chain to associate a user or users with a document) may also be sensitive. A query for the source of a piece of data may be recursively executed on provenance records to generate the chain. By appropriate access control policies on message records, we can easily achieve protection with different granularity on the chain based on the specific protection requirements:

- If we need to disclose the original sources without disclosing details in the chain, e.g. operations and actors, to a specific subject, we can address this requirement by only allowing the subject to access the *Source ID* and *Destination ID* of relevant message records.
- If we need to hide the actor information in the chain from a specific subject, we can deny the subject access to the *Actor ID* in the records in the chain.

7 Related Work

The proposed access control language has been influenced by the XACML language [24]. One distinct feature of our provenance access control is the need for the aggregation of authorization decisions from different policies with different purposes, e.g. organizational policies, user preferences with different versions, and privacy regulations. Because XACML does not distinguish between conditions and restrictions, XACML is not suitable for dealing with the aggregation required by the management of provenance. In addition its rule evaluation truth table and policy combining algorithms have several shortcomings [25]. The purpose handling has been inspired by the Privacy-aware Role-based Access Control model [23], however, it is more flexible than that in this earlier approach.

Security issues in the context of provenance management have been only briefly discussed in a few prior papers [3, 2, 9, 10, 1]. Groth et al. [3, 9] discussed security requirements for Service Oriented Architectures and proposed some abstract frameworks providing security mechanisms, including access control, for provenance stores. Braun and Shinnar [2], in the context of the PASS (Provenance-aware Storage Systems) project [26], discussed a security model for provenance, which consists of two separate models: one for protecting the structure or workflow (i.e., which ancestors and descendants are accessible to which users) and the other for specifying which node attributes are accessible to which users. Braun et al. [10] later argued the need for new security models for provenance management. In particular, they highlighted two properties, "DAG-nature" and "Immutability", of provenance information which distinguish provenance information from traditional data items and from tree-structured data. Hasan et al. [1] discussed research challenges to secure a provenance chain, and proposed a lifecycle model for provenance. They also analyzed possible applications of secure provenance. Compared to such work, our work is the only one focusing on the analysis of requirements for provenance access control and providing a comprehensive access control language to meet these requirements.

Very recently, the problem of secure provenance management has also been investigated in the broader context of *information networks* which abstract distributed information sharing [27, 28]. With respect to the domains that have been investigated and mentioned above (e.g., scientific database, grid computing, workflow, health care applications), this new problem domain introduces new challenges. For example, provenance in scientific databases keeps the modifications that have been applied to a specific data item, whereas workflow systems operate within the boundary of a single enterprise. In contract, information networks capture the movement/processing of data beyond any single database or enterprise, which means that each node can maintain a provenance store for all the data messages that have been received and that the provenance store is maintained by all the nodes in an information network that can cross many enterprises. The access control language presented in this paper can serve as a useful mechanism in this framework.

8 Conclusion and Future Work

Our proposal for provenance access control is still at its first stage, and many interesting problems are left open.

In the evaluation of provenance access control policies, decisions with uncertainties about the result of target evaluation or condition evaluation may arise. There are at least two cases in which a policy evaluation may generate uncertainties. First, because the predicates in a policy may refer to the content of other data or other provenance store or system variables, it might not be able to evaluate them due to the lack of privileges. There are several different design choices to address the issue of which privilege is needed for policy evaluation: the

administrative privilege, the query issuer privilege, and the policy author privilege. Obviously the administrative privilege, which may give excessive power to all table owners (they are policy authors), may result in severe security breaches. Rosenthal et al. [29] suggest that policies should be evaluated under the privilege of the query issuers rather than the policy authors. In contrast, Olson et al. [30] suggest that policies should be evaluated under the privilege of the policy authors rather than the query issuers. In either approach, it is possible that predicates in policies cannot be successfully evaluated due to the lack of privileges. Second, external factors, such as software vulnerabilities or hardware failures, may prevent predicates from being evaluated correctly as well. In both situations, uncertain decisions (neither permit or deny), in which we do not know the exact decision, are inevitable. The D-algebra [25] can be applied to deal with policy evaluation in the presence of uncertainty.

Delegation of access control rights, which is one important requirement for provenance access control [3, 10], has not been addressed in this paper. We prefer policy-based delegation management and consider delegation management policies to be meta-policies on access control policies and will investigate this issue in our future work.

Because of the semantics of different effects and predicates used in conditions and restrictions, inappropriate policy specifications may generate conflicting policies or redundant policies [31]. Detecting these abnormal policies is essentially a SAT problem. Fortunately, the problem size is usually very small regardless of the number of policies. Only policies with overlapping target spaces and sharing variables in predicates need to be checked. Various heuristic techniques have already been developed [31]; we need a tailored version for provenance access control policies as well.

Acknowledgement. This work is supported in part by AFOSR MURI award FA9550-08-1-0265.

References

[1] Hasan, R., Sion, R., Winslett, M.: Introducing secure provenance: problems and challenges. In: Proceedings of the 2007 ACM Workshop on Storage Security And Survivability (StorageSS), pp. 13–18 (2007)
[2] Braun, U., Shinnar, A.: A security model for provenance. Technical Report TR-04-06, Harvard University Computer Science (January 2006)
[3] Groth, P., Jiang, S., Miles, S., Munroe, S., Tan, V., Tsasakou, S., Moreau, L.: An architecture for provenance systems. Technical report, University of Southampton (November 2006)
[4] Benjelloun, O., Sarma, A.D., Halevy, A.Y., Theobald, M., Widom, J.: Databases with uncertainty and lineage. VLDB J. 17(2), 243–264 (2008)
[5] Buneman, P., Chapman, A., Cheney, J.: Provenance management in curated databases. In: SIGMOD 2006, pp. 539–550 (2006)
[6] Chapman, A., Jagadish, H.V., Ramanan, P.: Efficient provenance storage. In: [32], pp. 993–1006

[7] Heinis, T., Alonso, G.: Efficient lineage tracking for scientific workflows. In: [32], pp. 1007–1018
[8] Moreau, L., Groth, P.T., Miles, S., Vázquez-Salceda, J., Ibbotson, J., Jiang, S., Munroe, S., Rana, O.F., Schreiber, A., Tan, V., Varga, L.Z.: The provenance of electronic data. Commun. ACM 51(4), 52–58 (2008)
[9] Tan, V., Groth, P., Miles, S., Jiang, S., Munroe, S., Tsasakou, S., Moreau, L.: Security issues in a soa-based provenance system. In: Moreau, L., Foster, I. (eds.) IPAW 2006. LNCS, vol. 4145, pp. 203–211. Springer, Heidelberg (2006)
[10] Braun, U., Shinnar, A., Seltzer, M.: Securing provenance. In: HotSec 2008 (2008)
[11] Moreau, L., Plale, B., Miles, S., Goble, C., Missier, P., Barga, R., Simmhan, Y., Futrelle, J., McGrath, R., Myers, J., Paulson, P., Bowers, S., Ludaescher, B., Kwasnikowska, N., den Bussche, J.V., Ellkvist, T., Freire, J., Groth, P.: The open provenance model (v1.01). Technical report, University of Southampton (2008)
[12] Foster, I.T., Vöckler, J.S., Wilde, M., Zhao, Y.: Chimera: Avirtual data system for representing, querying, and automating data derivation. In: SSDBM, pp. 37–46. IEEE Computer Society, Los Alamitos (2002)
[13] Janee, G., Mathena, J., Frew, J.: A data model and architecture for long-term preservation. In: Larsen, R.L., Paepcke, A., Borbinha, J.L., Naaman, M. (eds.) JCDL, pp. 134–144. ACM, New York (2008)
[14] Callahan, S.P., Freire, J., Scheidegger, C.E., Silva, C.T., Vo, H.T.: Towards provenance-enabling paraview. In: Freire, J., Koop, D., Moreau, L. (eds.) IPAW 2008. LNCS, vol. 5272, pp. 120–127. Springer, Heidelberg (2008)
[15] Buneman, P., Khanna, S., Tan, W.-C.: Why and where: A characterization of data provenance. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001, vol. 1973, pp. 316–330. Springer, Heidelberg (2001)
[16] Abrams, M.D., Smith, G.W.: A generalized framework for database access controls. In: DBSec., pp. 171–178 (1990)
[17] McCollum, C.D., Messing, J.R., Notargiacomo, L.: Beyond the pale of mac and dac-defining new forms of access control. In: IEEE Symposium on Security and Privacy, pp. 190–200 (1990)
[18] Park, J., Sandhu, R.S.: Towards usage control models: beyond traditional access control. In: SACMAT, pp. 57–64 (2002)
[19] Park, J., Sandhu, R.S.: Originator control in usage control. In: POLICY, pp. 60–66. IEEE Computer Society, Los Alamitos (2002)
[20] Ni, Q., Bertino, E., Lobo, J.: An obligation model bridging access control policies and privacy policies. In: Ray, I., Li, N. (eds.) SACMAT, pp. 133–142. ACM, New York (2008)
[21] Raub, D., Steinwandt, R.: An algebra for enterprise privacy policies closed under composition and conjunction. In: Müller, G. (ed.) ETRICS 2006. LNCS, vol. 3995, pp. 130–144. Springer, Heidelberg (2006)
[22] Barth, A., Datta, A., Mitchell, J.C., Nissenbaum, H.: Privacy and contextual integrity: Framework and applications. In: IEEE Symposium on Security and Privacy, pp. 184–198. IEEE Computer Society, Los Alamitos (2006)
[23] Ni, Q., Trombetta, A., Bertino, E., Lobo, J.: Privacy-aware role based access control. In: Lotz, V., Thuraisingham, B.M. (eds.) SACMAT, pp. 41–50. ACM, New York (2007)
[24] Moses, T., ed.: eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Open (February 2005)
[25] Ni, Q., Bertino, E., Lobo, J.: D-algebra for composing access control policy decisions. In: ASIACCS (2009)

[26] Muniswamy-Reddy, K., Holland, D., Braun, U., Seltzer, M.: Provenance-aware storage systems. In: Proceedings of the 2006 USENIX Annual Technical Conference, pp. 43–56 (2006)
[27] Xu, S., Ni, Q., Bertino, E., Sandhu, R.: A characterization of the problem of secure provenance management. In: Workshop on Assured Information Sharing, Affiliated with the 2009 IEEE Intelligence and Security Informatics, ISI 2009 (2009)
[28] Xu, S., Sandhu, R., Bertino, E.: Tiupam: A framework for trustworthiness-centric information sharing. In: Third IFIP WG 11.11 International Conference on Trust Management, TM 2009 (2009)
[29] Rosenthal, A., Sciore, E.: Abstracting and refining authorization in sql. In: Jonker, W., Petković, M. (eds.) SDM 2004. LNCS, vol. 3178, pp. 148–162. Springer, Heidelberg (2004)
[30] Olson, L.E., Gunter, C.A., Madhusudan, P.: A formal framework for reflective database access control policies. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM Conference on Computer and Communications Security, pp. 289–298. ACM, New York (2008)
[31] Ni, Q., Lin, D., Bertino, E., Lobo, J.: Conditional privacy-aware role based access control. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 72–89. Springer, Heidelberg (2007)
[32] Wang, J.T.L. (ed.): Proceedings of the ACM SIGMOD International Conference on Management of Data. In: Wang, J.T.L. (ed.) SIGMOD 2008, SIGMOD Conference, Vancouver, BC, Canada, June 10-12, ACM, New York (2008)