IDIS (2010) 3:639–656
DOI 10.1007/s12394-010-0069-4

A persistent data tracking mechanism for user-centric identity governance

Hidehito Gomi

Received: 3 March 2010 / Accepted: 9 May 2010 / Published online: 17 July 2010

Abstract Identity governance is an emerging concept for fine-grained conditional disclosure of identity information and enforcement of corresponding data handling policies. Although numerous technologies underlying identity management have been developed, people still have difficulty obtaining a clear picture of how their identity information is maintained, used, and propagated. An identity management framework is described for tracking the history of how a person's identity information is handled after it is transferred across domains of control and for enforcing meta-policies related to managing identity information distributed over the Internet. With this framework, organizations that manage identity information can improve accountability for their data practices and thereby increase their trustworthiness. The framework also enables users to control and optimize the propagation of their identity information in a user-centric manner.

Keywords Identity governance · Policy enforcement · Privacy protection · User-centricity

Introduction

Identity management is based on technologies supporting electronic interactions requiring identity information. For example, identity federation technology is used to associate an individual's partial identities stored in disparate administrative domains on the Internet (OASIS 2005a; OpenID Foundation 2007). This technology enables a person to enhance the management of several

H. Gomi (✉)
Yahoo! JAPAN Research, 9-7-1 Akasaka, Minato-ku, Tokyo 107-6211, Japan
e-mail: hgomi@yahoo-corp.jp

partial identities spread over multiple systems. Other technologies include access control methods for granting or denying access to identity information from a different security domain. An important aspect of these technologies is that the interacting entities are distributed beyond domain boundaries and administered by different organizations. Thus, privacy protection is a key issue in identity management.

An emerging concept in identity management is *identity governance* (Liberty Alliance Project 2008). It addresses fine-grained conditional disclosure of identity information and enforcement of policies for handling identity information. Although there is not yet a uniform definition of the scope of identity governance in the technology community, identity governance overlays the supporting technologies and effectively establishes agreement between business partners handling identity information for compliance and audit reasons.

A recently developed concept in identity management is *user-centricity*, which means enabling a person to control his or her identity information and the way it is used (Bhargav-Spantzel et al. 2007). Although the idea itself is not new, it has been accepted as a basic guideline for constructing a system supporting strong user control and privacy protection. In this sense, since identity governance essentially requires users' involvement with transactions related to their identity information, it is desirable that the basic properties of user-centricity are applied to identity governance systems. A person should ideally understand all the states of his or her identity information scattered over the Internet and be able to take appropriate actions regarding them.

Unfortunately, it is very difficult for a person to get a clear picture of how his or her identity information is maintained, used, and propagated. This can be illustrated in a use case. Consider a motivating scenario about a patient and his or her medical information. Although a patient is generally required to provide personal data and informed consent before receiving medical treatment, it is difficult for the patient to confirm that the data is used in accordance with the consent provided. If medical personnel share the patient's data and clinical history with another medical facility providing additional service to the patient, the patient generally does not have a way to find out how the data is used by the second facility. In addition, medical facilities generally do not have sufficient means for demonstrating that they handle patient data appropriately.

In light of these considerations, we need a vision and a technical framework that puts people in control of their identity information and that optimizes identity propagation, including the types of information propagated, the organizations that are allowed to receive the information, and the policies for managing the information. The framework should encompass the broad perspective of the Internet. Although numerous supporting technologies have been developed for emerging standardization activities such as identity federation and personal attribute exchange, there has been little research on identity governance across domain boundaries from the user's viewpoint.

In this paper, an identity management framework is described that enables users to determine the current status and to track the usage history of their

identity information after it has been propagated from one trusted entity to another across domains of control. Also described are a scheme comprising several algorithms that enables users to ensure that their identity information is handled and propagated in an acceptable way and a mechanism for enforcing identity management policies.

The rest of this paper is organized as follows. Section "Model" introduces a model of the relationship between identity information and policy. Section "Transfer link management" describes the algorithms for managing and tracking the use of identity information. Section "Enforcement of identity management policies" describes the enforcement of identity management policies and constraints. Section "Discussion" discusses several issues related to identity governance. Section "Related work" describes related work, and Section "Conclusion and future work" concludes the paper with a summary of the key points and a look at future work.

Model

In this section, a model is described for managing identity information and tracking information practices.

A **data subject (DS)** is an individual to whom personal data relates. A DS delegates the secure management and effective utilization of his or her personal data to other entities, specifying preferences governing how the data is to be handled by them.

A **data manager (DM)** is an entity that manages the personal data of DSs on their behalf in accordance with their privacy preferences and in compliance with its own privacy and security policies. It is responsible for securely providing other entities with personal data on behalf of a DS. Even after a DS's data has been propagated to even more entities, the DM identifies their information practices in accordance with requests by the DS.

A **data consumer (DC)** is an entity that uses a DS's personal data obtained from a DM. This entity uses the data in a way that conforms to the agreement reached with the DM on how the data is to be used. It reports to the DM information on data usage and retention when relevant events have taken place in order to demonstrate its trustworthiness.

The above definitions of DM and DC are respectively related to the *data controller* and *data processor* definitions commonly used in the data protection context. DM and DC are different in that a DC has is responsible for handling personal data in compliance with the agreement it has with the DM whereas a data processor does not have this responsibility in definitions such as that of the UK's Data Protection Act.

The DM and DC are not actual entities; they are simply roles in the model. Therefore, a single entity can play both roles; that is, a DC entity can act as a DM entity once it receives and stores personal data. It is assumed that these entities are trusted and that they comply with the agreement. The purpose of

this work is to identify the information practices of trusted entities rather than to identify their misbehavior.

Capsuling of identity and policy

In this model, a DM manages personal data in accordance with policies that dictate how it is to be managed. If the DM needs to provide a DC with personal data, the DM encapsulates the data and related policy and transfers both to the DC.

The DC complies with the policy agreed upon and received from the DM prior to the data reception. This tight coupling of personal data and management policy prevents the DM from disclosing data to untrusted entities and enables it to control the data practices of DCs even after they receive the data.

The policies in the model have several aspects. The policies for DMs represent privacy constraints and conditions for how personal data is to be handled.

- Purpose: The context or the ways in which the data may be used: e.g., "my health care information may be used only for my treatment, not for research."
- Recipient: The person or entity with whom the data may be shared: e.g., "my medical records may not be shared with hospital H_0."
- Retention: How long the data may or must be kept: e.g., "my medical records must be kept for eight years after my last visit."
- Tracking: A report about how the data has been used or maintained, which is addressed in this work: e.g., "I need a report covering the last three years from hospital H_1 about how my immunization records were managed and which organizations used the data."

These constraints and conditions are requirements or obligations that need to be satisfied or fulfilled by the DCs after receiving data.

Dynamic composition of data and policy

When a DM and a DC interact during the transfer of a DS's data, they need to evaluate the associated risk and cost. The DM is concerned that, if the DC uses the data improperly, it will be accused of failing in its responsibility to protect the data. The DC is concerned that, if it receives unnecessary confidential data, the cost of data management and the risk of data leakage will be higher.

Therefore, prior to the transfer, the DM and DC negotiate and agree upon a policy for handling and using the data. Note that policy negotiation, matching, and conflict resolution mechanisms are not within the scope of this paper. Here, if their policies do not match, data transfer simply does not take place. We refer to the policy they agree upon as the *agreed upon policy*.

A key aspect of this model is the mechanism used to dynamically identify the minimal amount of personal data needed and compose the corresponding

policy. For restricting data disclosure, only those attributes of the DS's personal data that satisfy the agreed upon policy are used to dynamically produce an attestation, which is referred to here as an *assertion*. The assertion can contain basic information, such as issuer and recipients, be electronically signed by the issuer entity to enable detection of tampering, and be encrypted and propagated in a secure communication channel.

Transfer link

To enable the DS's data to be managed and controlled after it is transferred, a scheme is introduced for maintaining the relationship between a DM and a DC. This relationship is referred to as an *identity transfer link* from the DM to the DC. This link indicates that the DM has propagated a DS's data to the DC following agreement on a policy. In other words, the DM is responsible for tracking the DC's usage of the DS's data whereas the DC is responsible for reporting to the DM on how the data has been used or is being maintained if it is so requested. In a graph formed by nodes representing entities and edges representing transfer links, from a DC's viewpoint, the DM is called a *parent*, for explanatory convenience. Likewise, from a DM's viewpoint, a DC is called a *child*.

Transfer link management

This section explains the fundamental operations of identity transfer and accompanying link management based on the model described in Section "Model".

Transfer link creation

When an entity issues and propagates an assertion about a DS to another entity, a transfer link for the assertion is newly created for the lifecycle management of the DS's data. A transfer identifier is allocated to the link for use in this management. A DM creating a new transfer link is referred to as the *initial DM*; it does not have a parent.

Figure 1 illustrates the situation in which a transfer link from entity p to entity q is created. As described in Section "Model", assertion transfer entity p acts as the DM, and entity q acts as a DC.

A set of assertions issued by entity p to entity q is referred to as $A(p, q) = \{a_0^{p,q}, a_1^{p,q}, \cdots\}$. The assertions issued by p may depend on their recipient,

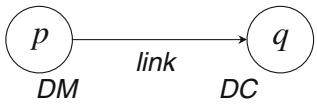

Fig. 1 Transfer link creation

q, because they may contain information about the DS that is private and confidential to p and q.

Algorithm 1 shows the procedure for a utility function that is used in algorithms explained later. When this function is called, a new transfer link object is created (step 1); the transfer identifier for the link, parent, *incoming assertion* (IA), and child data is set (steps 2–5). After a new store for *outgoing assertions* (OA) is created, a new assertion is added to the OA, and the function returns the new link (steps 6–8).

Algorithm 1 `new_transLink`(x, y, a, b, id)

1: $l \leftarrow$ `new_link()`
2: $l.trans\_id \leftarrow id$
3: $l.parent \leftarrow x$
4: $l.IA \leftarrow a$
5: $l.child \leftarrow y$
6: $l.OA \leftarrow$ `new_array()`
7: $l.OA.add(b)$
8: **return** l

Algorithm 2 shows the procedure for transfer link management for p when p propagates an assertion to q. It is assumed that p and q have an agreed upon policy for exchanging a set of specific data.

Algorithm 2 `create_transLink`(p, q)

Require: $p \neq q$, p and q have an agreed upon policy.
1: $a^{p,q} \leftarrow$ `create_assertion()`
2: $a^{p,q}.trans\_id \leftarrow$ `generate_id()`
3: $a^{p,q}.issuer \leftarrow p$
4: $a^{p,q}.policy \leftarrow$ `create_policy`(p, q)
5: `sign`$(a^{p,q})$
6: $l \leftarrow$ `new_transLink`$(null, q, null, a^{p,q}, a^{p,q}.trans\_id)$
7: $L(p).add(l)$
8: `send_assertion`$(q, a^{p,q})$

The $a^{p,q}$ and l denote an assertion and a transfer link from p to q, respectively; $x.(\cdot)$ indicates x's attribute information. The $a^{p,q}$ is produced using information about the new transfer identifier, the issuer, and the agreed upon policy obtained through policy negotiation between p and q (steps 1–4). The $a^{p,q}$ is signed by p, a transfer link is created, and its attribute information is set using Algorithm 1 (steps 5–6). In this case, the link does not have a parent since $a^{p,q}$ is originally issued on the basis of the stored personal data managed by p whereas the child is a recipient of assertion q. The link is registered in $L(p)$, which is a link table managed by p, and then $a^{p,q}$ is sent from p to q (steps 7–8).

Table 1 shows an example link table for DM p. It contains one transfer link and shows the transfer identifier, parent, incoming assertion, child, and outgoing assertions.

Table 1 Link table used by DM p to receive and send assertions

Trans_ID	Parent	Incoming assertion	Child	Outgoing assertions
123	null	null	q	$\{a^{p,q}\}$

Once q has received $a^{p,q}$ from p, it begins acting as a DC for p; the details of the procedure are omitted here (see Algorithm 6 in Section "Cyclic link detection"). The DC entity manages a transfer link table like the one shown in Table 2: q registers the incoming assertion, $a^{p,q}$, its transfer identifier, which is associated with $a^{p,q}$, and parent.

Transfer management revocation

A DM managing a transfer link cannot delete the link at its own discretion. However, events at its parent or child, such as data update and deletion, can trigger the DM to delete the corresponding link and send a link revocation request to other entities.

Algorithm 3 shows the procedure for transfer management revocation for DM p with a link to child q for assertion $a^{p,q}$ after a data update or deletion has taken place.

Algorithm 3 `revoke_mgt`$(p, q, a^{p,q})$

Require: $\exists l \in L(p); l.child = q, a^{p,q} \in l.OA$

1: $l.OA.del(a^{p,q})$
2: **if** $l.OA = \{\phi\}$, **then**
3: $\quad L(p).del(l)$
4: **end if**
5: `send_revokeRequest`$(q, l, a^{p,q})$

First, p finds and deletes $a^{p,q}$ from the list of transfer links $L(p)$ (step 1). Then, if the list has no assertion information (step 2), the list is deleted from $L(p)$ (step 3). Finally, p sends a message informing child q that the management of link l has ended (step 5). The description of this protocol is omitted here.

The timing of a DM performing this revocation procedure depends on its management policy. For instance, even if the DM receives a report from a DC about the deletion of data, the DM may keep the corresponding transfer link for a predefined duration and possibly request a check on data existence at the DC (See Section "Enforcement of identity management policies").

Table 2 Link table used by DC q to receive assertions

Trans_ID	Parent	Incoming assertion	Child	Outgoing assertions
123	p	$a^{p,q}$	null	null

Entity q receiving such a revocation request from parent entity p would follow the procedure described by Algorithm 4. The following link management operation is performed after the revocation.

Algorithm 4 `receive_revokeReq`$(p, q, a^{p,q})$

Require: $\exists l \in L(q); l.parent = p, l.IA = a^{p,q}$

1: **if** policy permits, **then**
2: $l.parent \leftarrow null$
3: $r \leftarrow l.child$
4: **for all** i such that $a_i^{q,r} \in l.OA$ **do**
5: $l.OA.del(a_i^{q,r})$
6: `send_revokeRequest`$(r, l, a_i^{q,r})$
7: **end for**
8: $L(q).del(l)$
9: **end if**

Algorithm 4 is based on the assumption that q finds an existing transfer link l related to assertion $a^{p,q}$ on the basis of the link information in the revocation request in Algorithm 3; $l.IA$ denotes an *incoming assertion* that corresponds to the link.

When a revocation request with assertion $a^{p,q}$ is received from p, the corresponding outgoing assertions are sought if the management policy of q permits the request (step 1). For the parent of q, the value *null* is assigned as the links that can be revoked (step 2). The child of q related to $a^{p,q}$ is r (step 3). The revocation operation is executed for each outgoing assertion $a_i^{q,r}$ (steps 4–7); i.e., $a_i^{q,r}$ is deleted from $l.OA$ (step 5). The revocation request for link l and $a_i^{q,r}$ is then performed recursively (step 6). After this is done for all outgoing assertions, l is deleted from the link table $L(q)$ (step 8).

Transfer Chaining

When a DC managing DS data obtained from a DM propagates the data to another entity, one more transfer link is newly created. Figure 2 illustrates the situation in which a new link between entities q and r is created given the situation illustrated in Fig. 1. Initially, q acts as DC for p, and then it acts as DM for r after the link chaining operation.

Algorithm 5 is a series of steps related to transfer link management by q when it propagates assertion $a^{q,r}$ to r. The assertion is produced using $a^{p,q}$ assuming that q has received $a^{p,q}$ from p and that there is a corresponding transfer link for $a^{p,q}$.

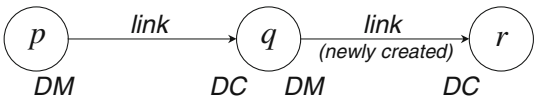

Fig. 2 Transfer link chaining

Algorithm 5 `chain_transLink`$(q, r, a^{p,q})$

Require: $\exists l \in L(q)$, $l.parent = p$, $l.IA = a^{p,q}$. Information to be sent from q to r is originally contained in $a^{p,q}$; $q \neq r$, $q \neq p$, and $r \neq p$.

1: **if** $a^{p,q}.policy$ permits agreed upon policy between q and r, **then**
2: $a^{q,r} \leftarrow a^{p,q}$
3: **else**
4: $a^{q,r} \leftarrow$ `create_assertion`$(a^{p,q})$
5: $a^{q,r}.issuer \leftarrow q$
6: $a^{q,r}.trans\_id \leftarrow a^{p,q}.trans\_id$
7: $a^{q,r}.policy \leftarrow$ `create_policy`$(q, r, a^{p,q}.policy)$
8: `sign`$(a^{q,r})$
9: **end if**
10: $l.child \leftarrow r$
11: $l.OA.add(a^{q,r})$
12: `send_assertion`$(r, a^{q,r})$

As Algorithm 5 shows, there are two cases for q sending an assertion to r. In one case, q forwards $a^{p,q}$ without any update to r. The assertion does not contain any private data or confidential information belonging to the DS that is shared only between p and q. Therefore, q and r agree on a data handling policy that is exactly the same as the policy attached to the original assertion, $a^{p,q}$, and the policy is simply passed from q to r (steps 1–2).

In the other case, q creates a new assertion based on information in the original assertion, $a^{p,q}$, issued by p (step 4). Although the issuer of the new assertion, $a^{q,r}$, is q, its transfer identifier is that of $a^{p,q}$ so that the transfer of $a^{q,r}$ is associated with $a^{p,q}$ (steps 5–6). The agreed upon policy for $a^{q,r}$ is newly created and set reflecting the original policy and policy negotiation between q and r, $a^{p,q}.policy$ (step 7). The $a^{q,r}$ is then signed by q (step 8). The child of q is set to the recipient entity, r, and $a^{q,r}$ is sent after it is registered in the existing list of outgoing assertions for the existing link (steps 10–12).

Cyclic link detection

It is possible for link paths among entities to form a cycle (*cyclic link*), as illustrated in Fig. 3. Detecting and avoiding cyclic paths is essential for proper link management.

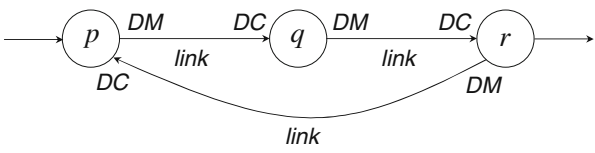

Fig. 3 Cyclic link paths

In the example shown in Fig. 3, the paths produced by transfer links p-q, q-r, and r-p form a cycle. In this case, p acts as a DM, providing q with a DS's data. It also receives the same information from r acting as a DC and DM. As a result, data operations could continue indefinitely or work inappropriately.

A transfer identifier in the model detects such links. Prior to propagating information, a DM checks that the recipient is not the entity itself or a parent that had propagated the information acting as a DM. That is, before q transfers information to r, it confirms that r is not p's parent. The procedures for an entity receiving an assertion from another entity, using a transfer identifier, are shown in Algorithm 6.

Algorithm 6 `receive_assertion`$(u, v, a^{u,v})$

Require: $\forall l_m \in L(u); \neg(l_m.trans\_id = a^{u,v}.trans\_id)$

1: **if** $\exists l \in L(v)$; $l.parent = null$, $l.child = w$, $a_0^{v,w} \in l.OA$, $l.trans\_id = a^{u,v}.trans\_id$, **then**
2: **if** data is latest and policy is satisfied, **then**
3: **if** $w = a^{u,v}.issuer$, **then**
4: `revoke_mgt`$(v, w, a_0^{v,w})$
5: **else**
6: `update_data`$(v, w, a_0^{v,w})$
7: **end if**
8: **else**
9: cancel the transfer
10: **end if**
11: **else**
12: $l \leftarrow$ `create_transLink`$(u, null, a^{u,v}, null, a^{u,v}.trans\_id)$
13: $L(v).add(l)$
14: **end if**

If v (possibly a DC) receives $a^{u,v}$ from u, it checks whether v itself is the original issuer of the assertion and is maintaining a corresponding transfer link as a DM for other DCs (step 1). If step 1 is true and if the data contained in the assertion is the latest (step 2), the role of v in relation to u is changed to DC because u acts as a DM although v was the initial DM. If w is the issuer of the received assertion, v sends a request to revoke the link to w to avoid link inconsistency using Algorithm 3 (steps 3–4).

Otherwise, v sends a data update request to w (step 6). Before the data is updated, its policy is also updated, u and w negotiate and agree upon policy on the basis of the original policy. A new assertion is created in `update_data`$(\cdot)$ containing the updated policy and sent to w. If v has no need to update its original data using the received data, it cancels the data transfer and, to avoid cyclic links, does not create a new link (steps 8–9). If there is no corresponding link related to $a^{u,v}$ (step 11), a new link is created; its transfer identifier, parent information, and incoming assertion $a^{u,v}$ are set in the link, and the link is added to link table $L(v)$ (steps 12–13).

Enforcement of identity management policies

This section describes the mechanism for enforcing policies that enable DSs to manage their identities with assistance from a DM. Once a DS has checked and verified his or her personal data, the DM acts appropriately to optimize the deployment of the data, enforcing the DS's pre-defined policies.

The identity management policies are rule-based and specified in a declarative language. The flexibility of logic representation enables DMs to easily incorporate DSs' preferences about how their personal data are to be used. The proposed rule-based policies take the following form:

$$h \Leftarrow b_1 \wedge b_2 \wedge \ldots \wedge b_m,$$

where h and b_i are *literals* that consist of respectively a head rule and body rules represented in first-order logic. The $\Leftarrow$ and $\wedge$ operators respectively specify reverse implication and logical conjunction of the rules. The head rule specifies an action for identity governance if the conditions of the corresponding body rules are satisfied. The body rules are constraints for triggering the head rule.

The following subsections focus on three types of identity management policies and their enforcement. These policies are at the meta-level and characterize the proposed model because the actions resulting from their enforcement invoke the transfer link management operations described in Section "Transfer link management". Since each action has a set of constraints as possible instances, additional constraints can be encoded into the original rules.

Reporting policies

A reporting policy specifies the rule to be followed by a DM upon receipt of a request from a DS for information about the practices being followed by the DM regarding the DS's personal data. A reporting rule for DS u takes the form:

$$\begin{aligned} doable(report(u, t_1, t_2, n, b, p)) \Leftarrow\ & lt(t_0 + itv, t) \wedge interval(u, itv) \\ & \wedge\ startTime(u, t_1) \wedge endTime(u, t_2) \\ & \wedge\ reportLevel(u, n) \\ & \wedge\ recursiveOpt(u, b) \\ & \wedge\ in(p, selectedEntities(u)). \end{aligned}$$

The axiom *doable* indicates that the action *report* with the specified arguments can and should be executed upon request if all the following constraints are satisfied.

- Frequency: A report is provided at specified periodic intervals (e.g., "monthly"). Function $lt(x, y)$ returns true if $x < y$. Parameters t_0 and t are

respectively the time when the latest report was produced and the current time. DS u specifies time interval constant itv by using $interval(u, itv)$.
- Duration: DS u specifies the duration for the report of events related to u by specifying $startTime(u, t_1)$ and $endTime(u, t_2)$.
- Report detail level: DS u specifies content level n for the report by specifying $reportLevel(u, n)$.
- Recursive reporting: DS u specifies whether the report should include sub-reports from all DCs to which the DS's personal data has been propagated by setting $recursiveOpt$ to either true or false.
- Entity selection: DS u specifies in his or her preferences the DMs that are to provide a report. Function $in(x, y)$ returns true if x is a member of y. DM p is included in the DMs specified in $selectedEntities(u)$.

Testing policies

Whereas the reporting described above is a comprehensive history for a specific interval, testing represents a snapshot of the specified data. It is done to determine whether a DC has been handling a DS's personal data appropriately. A testing policy specifies the rule to be followed by DCs when providing the current status of propagated personal data.

$$doable(testing(u, p, a)) \Leftarrow in(a, selectedAssertions(u)) \\ \wedge\ equals(recipient(a), p).$$

Action *testing* with the specified arguments can and should be executed upon request if the following constraint is satisfied.

- Assertion selection: DS u specifies the assertions for which he or she wishes the status to be tracked. Assertion a is included in the assertions specified in $selectedAssertions(u)$. Function $equals(x, y)$ returns true if $x = y$. Function $recipient(a)$ returns the identifier of the recipient of assertion a.

A DM receives a response as a result of the testing operation using the above policy and checks whether the result is consistent with the expected one. For example, if the DM expects that an assertion has been deleted at a DC, but the testing result from the DC shows that it still has the data for the assertion, the DC determines that the DC has handled the data inappropriately and either warns the DC to delete the data or degrades the trustworthiness of the DC.

Tracking termination policies

A tracking termination policy specifies the rule to be followed when terminating tracking activities such as reporting and testing by DMs and DCs. It specifies how long a DS has had a relationship with them, that the DS wants to know their practices regarding the DS's personal data, and under what

conditions the relationship is to be terminated. These specifications depend on the DS's preferences and may cover many possible criteria.

A tracking termination policy is a meta-policy incorporating both reporting and testing policies. If it is executed, the corresponding reporting and testing policies are revoked. This triggers revocation of the corresponding transfer links. Transfer links from a DM to a DC remain until the DS specifically revokes them at the DM since they are needed for tracking the corresponding personal data status even after the data has been deleted from the DC (see Section "Transfer management revocation"). For example, in the following tracking termination rule specified by a DS at a DM,

$$\begin{aligned} doable(terminateTracking(u, p, a)) \Leftarrow\ & gt(trustVal(p), v_t(u)) \\ \wedge\ & lt(riskVal(u, a), v_r(u)) \\ \wedge\ & in(a, assertions(u, p)) \\ \wedge\ & lt(t_{ex}(u, a), t), \end{aligned}$$

action *terminateTracking* triggers transfer link revocation with the defined arguments if the following constraints are satisfied:

- Trust evaluation: DS u does not require a DC to be tracked as a well trusted one if its degree of trustworthiness, represented by a trust metric calculated by the DM, is greater than threshold value $v_t(u)$ specified by u. Function gt returns true if $x > y$. $trustVal(p)$ denotes the calculated trust value for DC p.
- Risk evaluation: This constraint is related to the risk inherent in data propagation. The degree of risk is quantitatively calculated from economic and privacy viewpoints. Function $riskVal(u, a)$ returns the risk value related to assertion a and DS u; $v_r(u)$ is a constant value specified by u. Assertion a is included in the list of assertions, $assertions(u, p)$, issued by the DM to p for u.
- Expiration: This constraint sets a due date regarding an assertion provided to a DC. Function $t_{ex}(u, a)$ returns the expiration date for a specified by u. Parameter t is the current time.

Although this rule is an instance of automating tracking termination by means of a set of decision-making predefined in accordance with the preferences of the DS, the DS can easily modify the preferences at any time. The DS can also choose not to automate some or all of the decision-making process but rather to take action manually at any point in time.

When the *terminateTracking* operation is executed, transfer link revocation is invoked in accordance with Algorithm 7.

DM p searches its link table for transfer links with empty parent information as root links (step 1). If the conditions of the corresponding body rules are satisfied for terminating the tracking of assertion $a^{p,q}$ (step 2), p invokes function `revoke_mgt` (Algorithm 3), in which a link revocation request is sent

Algorithm 7 `call_linkRevocation`$(u, p, q, a^{p,q})$

1: **for all** i such that $l_i \in L(p); l.parent = null, l.OA = a^{p,q}$ **do**
2: **if** $doable(terminateTracking(u, q, a^{p,q})) = true$, **then**
3: `revoke_mgt`$(p, q, a^{p,q})$
4: **end if**
5: **end for**

to DC q (step 3). This causes q to update the corresponding link information in its link table in accordance with Algorithm 4 and its own tracking termination policies.

Discussion

This section discusses several topics and issues related to the proposed model and framework.

Trust

The proposed model assumes that trusted entities act in conformity with their agreements in a collaborative fashion. Since personal data can be potentially misused once it is disclosed to an entity if it is not encrypted even if the entity has agreed on a policy for its usage, detecting entity misbehavior is an open issue. An alternative for this issue is analysis of testing and reporting operations with various types of parameters to find inconsistency in DCs' reports.

However, the proposed approach is simply a foundation for improving identity governance because many current applications do not support the reporting of personal data usage and because the exchange of personal data between untrusted entities is unrealistic in practical business situations. The proposed framework is effective for managing trust because it provides a means for each participating entity to manage and update the trustworthiness of other entities if trustworthy reports are accumulated from the same entity. In addition, if data leakage takes place, the tracking results can be used to investigate and identify the cause of a misuse event and to determine where responsibility for it lies.

Design and implementation

In a high-level system architecture for the proposed model, functions of transfer link management, data reporting, and policy management are necessary for both DMs and DCs in addition to core identity management functions such as identity federation and attribute exchange.

The transfer link management operations are incorporated into assertion creation and propagation operations, as evidenced by the algorithms in Section "Transfer link management". The overhead for the link operations is limited although an overall implementation and feasibility study of the proposed framework remains for future work. The operations for revoking transfer links can be done asynchronously with the assertion operations to reduce response time overhead. The policy management function needs a logic rule engine for managing personal data that reflects a DS's privacy preferences and resolves policy conflicts between entities.

Trackability

Although some may feel that a DM does not need to monitor the behavior of a trusted DC, such monitoring provides a DS with the means to determine the current status and track the usage history of his or her identity information after it has been propagated. Although dependence on the trust relationship may reduce communication costs related to reporting, such dependence probably would not offer a sense of security to the DS.

However, this trackability may lead a DM to accumulate personal data about a DS. In the proposed model, a transfer identifier is associated with every data transfer containing personal data. The identifier is a dynamic and transient one that covers limited actions related to the usage of personal data.

Accountability

Entities are at risk of involvement in inappropriate data handling or disclosure if problems such as unauthorized data disclosure are made known. Since entities in different domains have different responsibilities related to data practices, using this framework is an effective way to demonstrate that their data handling actions have complied with the agreed upon policies. The tracking and reporting schemes used in the proposed framework enable entities to improve accountability for their identity protection practices.

Auditing is an alternative method for entities to demonstrate the correctness of their protection practices. In many cases, auditing authorities monitor the identity management log for legal or compliance reasons in a centralized manner. In contrast, the proposed framework takes a decentralized approach—a DS can ask a DM to aggregate the data usage reports of its children.

Policy combination and conflict resolution

The three entities (DS, DM, DC) in this model have either policies or preferences that need to be unified across all three entities to enable operations using personal data although there may be conflicts among the policies and preferences. A DS's preferences regarding his or her personal data generally take precedence over the DM's privacy policy.

However, if data related to a DS is security-related and not owned by the DS, the DM's own policies may take precedence. For example, although information about a privilege for enjoying particular digital content is related to a DS, the administrator may wish to enforce a policy that only the DS can exercise the privilege to prevent the content's unauthorized distribution. There are several open issues like this that need to be resolved before the proposed framework can be completely implemented.

Related work

There has been much research related to privacy and identity management. This research is built on such technologies as identity federation and personal attribute exchange.

The idea of controlling access to data even after it is disseminated has been considered, especially in the area of digital rights management (Schneck 1999). Casassa Mont et al. (2003) introduced a concept called "sticky policies"—a handling policy is attached directly to a package of data. This is similar to the basic idea of the model presented here—creating a tight association between data and a policy for controlling its use. In their concept, a third party is responsible for auditing policy enforcement whereas, in the approach described here, the entity originally responsible for managing and propagating the data retains control over the information practices of data users and aggregates usage data in a decentralized manner.

There has been much work on privacy policy description languages and constraints. The Liberty Identity Governance Framework (IGF) (Liberty Alliance Project 2008) specifies privacy constraints. P3P (W3C 2002b) and its complement, APPEL (W3C 2002a), provide means for expressing comprehensive user preferences. XACML (OASIS 2005b) specifies an access control language and privacy extensions. Ahn and Lam (2005) proposed using a user preference expression language in federated identity management systems. Zhang et al. (2003) presented a rule-based framework in which users' policies are represented in a declarative language.

Privacy enhancing technologies and privacy management are important topics for identity management. Squicciarini et al. (2007) proposed providing a new set of assertions to define the privacy related properties of a federation system. Paci et al. (2009) proposed an approach for the controlled release of identity attributes supporting selective and incremental credential disclosure.

The PRIME project (Leenes and Schallaböck 2008) provides a data tracking functionality that can be used to obtain information on the status of data after it is released. This transparency feature as well as the focus on managing policies, preferences, and data from a user's viewpoint are relevant to the work described here. Similarly, the PrimeLife Project (2009) and EnCoRe Project (2010) aim at dealing with privacy management and users' preferences.

Baldwin and Shiu (2003) described an audit service that acts as a central place for logging data received from other IT systems. In contrast, the proposed model takes a decentralized approach in which data use reports from multiple entities are integrated in accordance with the order of the data propagation.

Mashima and Ahamad (2008) described a system that monitors the usage of personal data and captures context information related to the requestor. Their work is relevant to that described here in that both deal with the monitoring of data usage. Their objective was to detect anomalous actions by impersonating malicious users whereas the objective here is to reveal how personal data is used and propagated from a privacy protection viewpoint.

Conclusion and future work

The model described in this paper supports lifecycle management of personal data across domain boundaries using identity transfer links. Algorithms for fundamental operations of link management enable determination of the current status of data after it has been propagated. The proposed framework enables users to control the propagation of their personal data in a user-centric manner. At the same time, it helps participating entities to improve accountability for their data practices. Future work includes further investigation of trust management and policy conflict resolution.

References

Ahn G, Lam J. Managing privacy preferences for federated identity management. In: Proceedings of the 2005 ACM workshop on digital identity management (DIM'05). 2005. p. 28–36.

Baldwin A, Shiu S. Enabling shared audit data. In: Proceedings of the 6th international security conference (ISC'03). 2003. p. 14–28.

Bhargav-Spantzel A, Camenish J, Gross T, Sommer D. User centricity: a taxonomy and open issues. J Comput Secur. 2007;15(5):493–527.

Casassa Mont M, Pearson S, Bramhall P. Towards accountable management of identity privacy: sticky policies and enforceable tracing services. In: Proceedings of the 14th international workshop on database and expert systems applications (DEXA'03). 2003. p. 377–82.

EnCoRe Project. Technical architecture for the first realized case study. 2010.

Leenes R, Schallaböck J. PRIME white paper. 2008. http://www.prime-project.eu/. Accessed 31 March 2010.

Liberty Alliance Project. Liberty IGF privacy constraints specification. 2008. http://www.projectliberty.org/specs. Accessed 31 March 2010.

Mashima D, Ahamad M. Towards a user-centric identity-usage monitoring system. In: Proceedings of the 3rd conference on internet monitoring and protection (ICIMP'08). 2008. p. 47–52.

OASIS. Assertions and protocol for the OASIS security assertion markup language (SAML) V2.0. 2005a. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security. Accessed 31 March 2010.

OASIS. eXtensible Access Control Markup Language (XACML). 2005b.

OpenID Foundation. OpenID authentication 2.0—final. 2007. http://openid.net/. Accessed 31 March 2010.
Paci F, Bauer D, Bertino E, Blough D, Squicciarini A, Gupta A. Minimal credential disclosure in trust negotiations. Identity in the Information Society; 2009.
PrimeLife Project. Privacy and identity management in Europe for life. 2009.
Schneck P. Persistent access control to prevent piracy of digital information. Proc IEEE. 1999; 87(7):1239–50.
Squicciarini A, Hintoglu A, Bertino E, Saygin Y. A privacy preserving assertion based policy language for federation systems. In: Proceedings of the 12th ACM symposium on access control models and technologies (SACMAT'07). 2007. p. 51–60.
W3C. A P3P preference exchange language 1.0 (APPEL1.0). 2002a. http://www.w3.org/TR/P3P-preferences/. Accessed 31 March 2010.
W3C. The platform for privacy preferences 1.0 (P3P1.0) specification. 2002b. http://www.w3.org/TR/P3P/. Accessed 31 March 2010.
Zhang L, Ahn G, Chu B. A rule-based framework for role-based delegation and revocation. ACM Trans Inf Syst Secur. 2003;6(3):404–41.