User-Centric Identity Governance across Domain Boundaries

Hidehito Gomi
Yahoo! JAPAN Research
9-7-1 Akasaka, Minato-ku, Tokyo 107-6211, JAPAN
hgomi@yahoo-corp.jp

ABSTRACT
Identity management is a set of viable technologies for supporting electronic interactions requiring identity information in the digital world. Although numerous elemental technologies have been developed in support of emerging standards and specifications, there has been little research on identity governance across domain boundaries from the user’s viewpoint. It is thus still difficult for users to understand how their own identity information is being maintained, used, and propagated. An identity management framework is described for tracing the history of how a user’s identity information is handled after it is transferred across domains of control. 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.

Categories and Subject Descriptors
K.6.5 [Management of Computing and Information System]: Security and Protection

General Terms
Management, Security

Keywords
Policy Enforcement, Identity Governance, Privacy

1. INTRODUCTION
Identity management is a set of viable technologies for supporting electronic interactions requiring identity information in the digital world. Since a person typically has various types of identifying information managed by disparate administrative entities depending on the context, such as work and home, it would be ideal if a person could control all his/her digital identities from a simplified interface by exchanging information among trusted organizations in a secure manner. However, since a digital identity may represent the person’s private information, it is necessary to appropriately protect his/her privacy no matter what situation he/she encountered. That is, there is a trade-off between usability and privacy in the digital world.

Unfortunately, it is very difficult for people to get a clear picture of how their own identity information is maintained, used, and propagated. To understand a use case, consider a motivating scenario about a patient and his/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. Moreover, it is difficult for the patient to transfer the data from a facility to a more trusted one if he/she wishes. In addition, medical facilities generally do not have sufficient means for demonstrating that they handle patient data appropriately.

In the light of these considerations, we need a vision and a technical framework that puts people in control of their identity data and that optimizes identity propagation, including the types of data propagated, the organizations managing the data, and the policies for managing the data. The framework should be based on the broad perspective of the Internet. Although numerous elemental technologies have been developed in support of 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 is a scheme comprising several algorithms that enables users to ensure that their identity information is handled and propagated in a privacy preserving manner.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 introduces a model of the relationship between identity data and policy. Section 4 describes the algorithms for managing and tracing of the use of identity data, and Section 5 describes an example of identity deployment optimization based on the model. Section 6
explains an information practices tracing mechanism that puts users in control of their identity information. Section 7 presents an example of policy binding and its description. Section 8 discusses several issues related to identity governance. Section 9 concludes the paper with a summary of the key points and a look at future work.

2. RELATED WORK

There has been much research and standardization related to privacy and identity management. This research is built on such technologies as identity federation and personal attribute exchange. There has been much less research on such technologies as identity federation and personal attribute exchange. There has been much less research on distributing identity information over the Internet in a user-centric fashion.

Mont et al. introduced a concept called “sticky policies” – a handling policy is attached directly to a package of data [7]. This is similar to the basic idea of the model presented here – a tight association between identity data and the policy for handling it. In their concept, a third party is responsible for auditing policy enforcement and compliance 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 before sending its report to the data owner.

The Liberty Identity Governance Framework (IGF) [10] specifies privacy constraints, which is relevant to this work if privacy policies are assumed to be matched prior to identity data exchange. While this framework focuses on the expressive description of privacy policies and preferences, the focus here is on the coalition of identity-based services to enable privacy enhanced propagation of identity data and the related policies.

There has been much work on privacy policy description languages. P3P [13] and its complement APPEL [12] provide means for expressing comprehensive user preferences. They are a good starting point for specifying privacy constraints. Ahn et al. proposed using a user preference expression language in federated identity management systems [11]. In the work described here, privacy policy description is a crucial topic because a policy for handling identity information is mutually agreed upon by the sender and receiver. It is used as a tool for enforcing the owner’s preferences.

Privacy enhancing technologies and privacy management are important topics for identity management. Gevers et al. introduced privacy enhanced claim uniform resource identifiers, which enable one to request personal data in a privacy friendly manner from existing identity management systems [3]. Squicciarini et al. proposed providing a new set of assertions to define the privacy related properties of a federation system [11]. Claubö et al. described a system that provides anonymity and accountability at the same time by maintaining the unlinkability of pseudonyms in a user-centric fashion [2]. Although privacy protection is also an important issue here, techniques are described for tracing identity information practices and for facilitating identity information exchange among trusted partners.

Regarding data tracking, the PRIME project [5] provides a data tracking functionality that can be used to obtain information on the status of the data after it is released. This transparency feature as well as the focus on managing policies, preferences, and data from the user’s viewpoint is relevant to the work described here. In addition, the work described here provides a technical approach to data tracking and optimization that can be adapted to existing identity exchange protocols and schemata in a collaborative way, even if there is a data distribution chain.

Mashima et al. described a system that monitors the usage of identity data and captures context information related to the requestor such as geolocation data [6]. Their work is relevant to that described here in that both deal with the monitoring of data usage. Their objective is to detect anomalous actions by impersonating malicious users whereas the objective here is to reveal how identity data is used and propagated from a privacy protection viewpoint, even if they are trusted.

3. MODEL

In this section, a model is described for governing identity information, tracing information practices, and promoting secure identity information exchange. First, several terms are defined.

A **data owner (DO)** is a person who owns his/her identity-related data. This person initially provides his/her identity data for various entities for secure management and convenient utilization of the data via a user agent. The person specifies preferences for how the data is to be used and handled by the various entities.

A **data manager (DM)** is an entity that stores and maintains the identity data of DOs on their behalf. This entity manages the identity data in compliance with its own privacy or security policies. It is responsible for securely providing other entities with the identity data on the DOs’ behalf. Even after the data has been provided and has been propagated to even more entities, the DM identifies their information practices in accordance with requests by the DOs. The DM thus provides an interface that the DOs can use to monitor, trace, and control their identity data.

A **data consumer (DC)** is an entity that uses DOs’ identity data obtained from DM entities. This entity uses the data in a way that conforms to the agreement reached with the DM on data usage. It reports to the DM information on data usage and retention when relevant events have taken place in order to demonstrate its own trustworthiness.

The DM and DC are not actual particular entities but 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 identity data.

3.1 Capsuling of Identity and Policy

In this model, a DM entity manages identity data associated with policies that dictate how it is to be managed. If the DM needs to provide a DC entity with the identity data, the DM encapsulates the data and policies 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 identity data and management policy prevents the DM from disclosing the data to untrusted entities and enables it to govern the data practices of DCs even after they receive the data.

The policies in the model have several aspects. The policies for DM entities represent privacy constraints and conditions for how identity data is to be handled. These policies have the following aspects in particular.
• Purpose: The context or the ways in which the data may be used: e.g., “my health care information is used only for my treatment, not for research.”
• Recipient: The person or entity with whom the data may be shared: e.g., “do not disclose my medical records to hospital A.”
• Authorization: Who can access the data: e.g., “my family can access all of my medical records.”
• 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.”
• Tracing: A report about how the data is used or maintained, which is characteristic of this work: e.g., “the hospital reported to me that my immunization records were disclosed to public health authorities for the purpose of controlling disease.”

These constraints and conditions are requirements or obligations that need to be satisfied or fulfilled by the DC entities after receiving data. Although the types of the policies of DC entities are only similar to these privacy constraints and conditions, they imply a DC entity’s commitment to privacy protection when presenting a DM entity with a request for data.

A DO has preferences about how his/her own data is to be handled by DM and DC entities. In this model, since a DO’s preferences and data are strongly associated, the DO’s preferences are respected by DC entities following data receipt.

3.2 Dynamic Composition of Data and Policy

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

Therefore, prior to the transfer, the DM and DC entities negotiate and agree upon policies for handling and using the data. Note that such policy negotiation and a policy-matching mechanism are not the main topics of this paper. Simply, if their policies do not match, data transfer does not take place. We refer to the policy they agree upon as the agreed upon policy.

In addition, the DO needs to be involved in decisions made regarding data transfer. Thus, the preferences of the DO are reflected in the DM entity’s policy since the DM is delegated the management role by the DO. It would thus be better to say that the policies and preferences of these three entities (the DM and DC entities and the DO) are used to ensure proper use of the data after it is transferred.

A key aspect of this model is the dynamic mechanism used to compose the minimal amount of identity data and the corresponding policy. For restricting data disclosure as a basic privacy preserving approach, only those attributes of the DO’s identity data such as purpose 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.

It is difficult to specify a static description for policies and preferences due to frequent changes in the trust criteria and risk and because the DO may not know his/her own potential wishes. In such a case, the DM entity needs to base its policy on the latest criteria and to obtain the DO’s consent at the time of need, at which time it presents helpful information to the DO.

3.3 Transfer Link

To enable the DO’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 entity. This relationship is referred to as an identity transfer link from the DM entity to the DC entity. This link indicates that the DM entity has propagated a DO’s data to the DC entity following agreement on a policy. In other words, when a link exists between a DM and a DC, the DM is responsible for tracing the DC’s usage of the DO’s data whereas the DC is responsible for reporting to the DM on how the data has been used or is maintained if it is so requested.

In a graph formed by nodes representing entities and edges representing transfer links, from a DC entity’s viewpoint, the DM entity is called a parent, for explanatory convenience. Likewise, from a DM entity’s viewpoint, a DC entity is called a child.

Figure 1 illustrates a transfer link representing an event in which a DM entity has transferred to a DC entity a capsule containing data and the related policy.

3.4 Architecture

A high-level system architecture was constructed in which the components satisfy the requirements described above and work together (Fig. 2).

The core identity management (IdM) part in the DM and DC includes fundamental functions of identity management such as user authentication, identity federation, and primitive attributes exchange, which are not the main topics of this paper. The transfer link management function in the DM and DC maintains the transfer link explained in Sec. 3.3.

In the DM, the trust management function stores information about trusted entities and evaluates the trustworthiness of entities and risks arising when interacting with them. The policy management function maintains the DM’s policies regarding data management and creates policies for constraining the DCs that reflect the DO’s preferences. It also checks the DCs’ policies, creates agreed upon policies, and stores them when data is transferred. The tracing function aggregates the data usage reports from the DCs. The data managing reporting function produces a log listing the DM’s data management actions.
In Assrtn Child {
ID initial DM a new transfer link is referred to as the link for use in the management. A DM entity creating an assertion about a DO to another entity, a transfer identifier is allocated gates an assertion for the lifecycle management for. When an entity issues and propagates an assertion, it assumes to exist. When an entity issues and propagates an assertion, it assumes to exist. When an entity issues and propagates an assertion, it assumes to exist. When an entity issues and propagates an assertion, it assumes to exist.

4. TRANSFER LINK MANAGEMENT

This section explains in more detail the fundamental operations of identity transfer and accompanying link management based on the model described in Sec. 3.

4.1 Transfer Link Creation

In this model, a set with a finite numbers of trusted entities is assumed to exist. When an entity issues and propagates an assertion about a DO to another entity, a transfer link for the assertion is newly created for the lifecycle management of the DO’s data. A transfer identifier is allocated to the link for use in the management. A DM entity creating a new transfer link is referred to as the initial DM; it does not have a parent.

Figure 3 illustrates the situation in which a transfer link from entity p to entity q is created. As described in Sec. 3, assertion transfer entity p acts as the DM, and entity q acts as a DC.

![Figure 3: Transfer Link Creation.](image)

A set of assertions issued by entity p to entity q is referred to as A(p,q) = {a_p, a_{p,q}, \ldots}. The assertions issued by p may depend on their recipient, q, because they may contain information about the DO that is private and confidential to p and q.

Algorithm 1 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 1 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 create_link()
7: l.trans_id \leftarrow a^{p,q}.trans_id
8: l.parent \leftarrow none
9: l.child \leftarrow q
10: l.OA \leftarrow create_array()
11: l.OA.add(a^{p,q})
12: L(p).add(l)
13: send_assertion(q, a^{p,q})

In Alg. 1, a^{p,q} and l denote an assertion and a transfer link from p to q, respectively; x(.) indicates x’s arbitrary information. From step 1 to 4, a^{p,q} is produced using the information about a new transfer identifier, an issuer, and an agreed upon policy. In step 5, a^{p,q} is signed by p. From step 6 to 11, a transfer link is created, and its attribute information is set. The transfer identifier for the link is set using that of a^{p,q} (step 7). In this case, the link does not have a parent (step 8) since a^{p,q} is originally issued on the basis of the stored identity data managed by p whereas the child is a recipient of assertion q (step 9). OA, which denotes a list of outgoing assertions sent to q, is newly created (step 10); a^{p,q} is registered in l.OA (step 11). The link is registered in L(p), which is a link table managed by p (step 12). Finally, a^{p,q} is sent from p to q (step 13).

Table 1 illustrates an example of a link table for DM entity 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 entity p to receive and send assertions**

<table>
<thead>
<tr>
<th>Trans_ID</th>
<th>Parent</th>
<th>InAsrtn</th>
<th>Child</th>
<th>OutAsrtn</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>none</td>
<td>none</td>
<td>q</td>
<td>{a^{p,q}}</td>
</tr>
</tbody>
</table>

Once q has received a^{p,q} from p, it begins acting as a DC for p; the details of the procedures are omitted here (see Alg. 5 in Sec. 4.3). The DC entity manages a transfer link table like the one shown in Tab. 2.

In Tab. 2, q registers the incoming assertion, a^{p,q}, its transfer identifier, which is associated with a^{p,q}, and parent.
Table 2: Link table used by DC entity q to receive assertions

<table>
<thead>
<tr>
<th>Trans_ID</th>
<th>Parent</th>
<th>InAssrtns</th>
<th>Child</th>
<th>OutAssrtns</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>p</td>
<td>a\textsuperscript{p,q}</td>
<td>none</td>
<td>none</td>
</tr>
</tbody>
</table>

4.2 Transfer Management Revocation

A DM entity 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 entity to delete the corresponding link and send a request to revoke the link management to other entities.

Algorithm 2 shows the procedure for transfer management revocation for DM entity p with a link to its child q for assertion a\textsuperscript{p,q} after a data update or deletion has taken place.

Algorithm 2 revoke_mgt(p, q, a\textsuperscript{p,q})

Require: \exists n \in N \mid l_n \in L(p), l_n.child = q, a\textsuperscript{p,q} \in l_n.OA
1: l_n.OA.delete(a\textsuperscript{p,q})
2: if l_n.OA = \{\} then
3: L(p).delete(l_n)
4: end if
5: send_revokeRequest(q, l_n, a\textsuperscript{p,q})

In Alg. 2, p finds and deletes a\textsuperscript{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 request to inform child q that the management of link l_n has ended (step 5); the description of this protocol is omitted here.

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

Entity q receiving such a revocation request from entity p, which is the parent of q, would follow the procedure described by Alg. 3. The following link management operation is performed after the revocation.

Algorithm 3 receive_revokeReq(p, q, a\textsuperscript{p,q})

Require: \exists n \in N \mid l_n \in L(q), l_n.parent = p, l_n.IA = a\textsuperscript{p,q}
1: if policy permits then
2: r ← l_n.child
3: for all i such that a\textsuperscript{i,r} \in l_n.OA do
4: l_n.OA.delete(a\textsuperscript{i,r})
5: send_revokeRequest(r, l_n, a\textsuperscript{i,r})
6: end for
7: L(q).delete(l_n)
8: end if

In Alg. 3, it is assumed that on the basis of the link information in the revocation request in Alg. 2, q finds an existing transfer link l_n related to assertion a\textsuperscript{p,q}; l_n.IA denotes an incoming assertion that corresponds to the link.

When a revocation request with assertion a\textsuperscript{p,q} is received from p, the corresponding outgoing assertions are sought if the management policy of q permits the request (step 1). The child of q related to a\textsuperscript{p,q} is r (step 2). For every outgoing assertion a\textsuperscript{i,r}, its revocation operation is executed; i.e., the a\textsuperscript{i,r} is deleted from l_n.OA (step 4). Then the revocation request for link l_n and a\textsuperscript{i,r} is performed recursively (step 5).

After this is done for all the outgoing assertions, l_n is deleted from the link table L(q) (step 7).

4.3 Transfer Chaining

When a DC entity managing DO data obtained from its DM entity propagates the data to another entity, one more transfer link is newly created (transfer chaining). Figure 4 illustrates the situation in which a new link between entities q and r is created given the situation illustrated in Fig. 3.

![Figure 4: Transfer Link Chaining.](image)

In Fig. 4, q originally acts as a DC for entity p, and then acts as the DM for r after the link chaining operation.

Algorithm 4 is a series of steps related to transfer link management by q when it propagates assertion a\textsuperscript{p,q} to r. The assertion is produced using a\textsuperscript{p,q}.

Algorithm 4 chain_transLink(q, r, a\textsuperscript{p,q})

Require: the information sent from q to r is originally contained in a\textsuperscript{p,q}; q ≠ r, q ≠ p, and r ≠ p.
1: if a\textsuperscript{p,q}.policy permits an agreed upon policy between q and r then
2: a\textsuperscript{q,r} ← a\textsuperscript{p,q}
3: else
4: a\textsuperscript{q,r} ← create_assertion(a\textsuperscript{p,q})
5: a\textsuperscript{q,r}.issuer ← q
6: a\textsuperscript{q,r}.trans_id ← a\textsuperscript{p,q}.trans_id
7: a\textsuperscript{q,r}.policy ← create_policy(q, r, a\textsuperscript{p,q}.policy)
8: sign(a\textsuperscript{q,r})
9: end if
10: if \exists n \in N \mid l_n \in L(q), l_n.trans_id = a\textsuperscript{p,q}.trans_id then
11: L.OA.add(a\textsuperscript{q,r})
12: else
13: l ← create_link()
14: l.trans_id ← a\textsuperscript{p,q}.trans_id
15: L.child ← r
16: L.OA.add(a\textsuperscript{q,r})
17: L(q).add(l)
18: L(q).add(l)
19: end if
20: send_assertion(r, a\textsuperscript{q,r})

As Alg. 4 shows, there are two cases for q sending an assertion to r. In one case, q forwards a\textsuperscript{p,q} without any update to r. The assertion does not contain any private or confidential information belonging to DO that is shared only between p and q. This case implies that q and r agree on a data handling policy that is exactly the same as the policy attached to the original assertion, a\textsuperscript{p,q}. Therefore, the policy is simply passed from q to r (steps 1 and 2).

In the other case, q creates a new assertion based on the information of the original assertion, a\textsuperscript{p,q}, issued by p (step 4). Although the issuer of the new assertion, a\textsuperscript{q,r}, is q (step 5), its transfer identifier is that of a\textsuperscript{p,q} so that the transfer of
$a^{\varphi,r}$ is associated with $a^{\nu,q}$ (step 6). The agreed upon policy for $a^{\varphi,r}$ is newly created and set reflecting the original policy, $a^{\rho,\text{policy}}$ (step 7). The $a^{\varphi,r}$ is then signed by $q$ (step 8).

From step 10 to 19, the transfer link management operations for chaining are executed. If there is an existing link with an identifier the same as that of $a^{\varphi,r}$ (step 10), $a^{\varphi,r}$ is registered in the outgoing assertions of the existing link information (step 11). Otherwise, a new transfer link is created, and appropriate information is set for it (from steps 13 to 18). The child of $q$ is set to the recipient entity, $r$ (step 15).

After a new list of outgoing assertions for the link is created (step 16), $a^{\varphi,r}$ is registered in the list (step 17), and then the link is registered in link table $L(q)$ (step 18). Finally, $a^{\varphi,r}$ is sent to $r$ (step 20).

Table 3 shows that the child and outgoing assertions are added to an existing list with transfer identifier $123$ in the link table managed by $q$ after transfer link chaining from $q$ to $r$ has taken place given the situation illustrated in Fig. 3 and Tab. 2.

<table>
<thead>
<tr>
<th>Trans_ID</th>
<th>Parent</th>
<th>InAssrtn</th>
<th>Child</th>
<th>OutAssrtns</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>$p$</td>
<td>$a^{\varphi,q}$</td>
<td>$r$</td>
<td>{ $a^{\nu,r}$ }</td>
</tr>
</tbody>
</table>

### 4.4 Cyclic Link Detection

It is possible for link paths among entities to form a cycle (cyclic link), as illustrated in Fig. 5. Detecting and avoiding cyclic paths is needed for proper link management.

![Figure 5: Cyclic Link Paths.](image)

In the example shown in Fig. 5, 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 DO’s data. It also receives the same information from $r$ acting as a DC and DM. As a result, operations such as data and policy updating and data usage reporting could continue indefinitely or work inappropriately.

A transfer identifier in the model detects cyclic links. Prior to propagating information, a DM entity checks that the recipient is not the entity itself or a parent that had propagated the information acting as a DM. Before $q$ transfers information to $r$, it confirms that $r$ is not parent $p$. The DC entity to receive the data checks the transfer identifier of the assertion received from the DM to see if it matches any other transfer identifier for the DO associated with the assertion. In the example here, $r$ sends an assertion to $p$, and $p$ checks whether the transfer identifier associated with the assertion matches one associated with any of the assertions entity $p$ has ever issued. In this case it would, so the data would not be propagated.

The procedure for detecting and avoiding cyclic paths is encapsulated in four algorithms. Algorithms 1, 4, and 6 (see Sec. 4.5) are used by entity $u$ when sending assertion $a^{u,v}$. Algorithm 5 is used by entity $v$ after it receives the assertion.

**Algorithm 5 receive_assertion($u$, $v$, $a^{u,v}$)**

**Require:** $\exists m \notin N; \ l_m \in L(u)$; $l_m.\text{trans}\_id = a^{u,v}.\text{trans}\_id$

1: if $\exists n \in N; \ l_n \in L(v)$; $l_n.\text{parent} = \text{none}$; $l_n.\text{child} = v$; $a^{u,v}_0 \in l_n.OA$; $l_n.\text{trans}\_id = a^{u,v}.\text{trans}\_id$ then
2: if data is latest and managing policy satisfies then
3: if $w = a^{u,v}.\text{issuer}$ then
4: $\text{revoke}_{\text{neg}}(v, w, a^{v,w}_0)$
5: else
6: $\text{update}_{\text{data}}(v, w, a^{v,w}_0)$
7: end if
8: else
9: cancel the transfer
10: end if
11: else \{there is no corresponding link\}
12: $l \leftarrow \text{create}_{\text{link}}()$
13: $l.\text{trans}\_id \leftarrow a^{u,v}.\text{trans}\_id$
14: $l.\text{parent} \leftarrow u$
15: $\text{IJA} \leftarrow a^{u,v}$
16: $L(v).\text{add}\(l)$
17: end if

The assumption of Alg. 5 differs from that of Alg. 6 in that the assertion sender in the former does not have a corresponding transfer link for the receiver whereas that in the latter does have one.

If $v$ (possibly a DC entity) receives $a^{u,v}$ from $u$, it checks whether the original issuer of the assertion is $v$ itself and maintains a corresponding transfer link as a DM for other DC entities (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 $v$ acts as a DC for maintaining links although $u$ was the initial DM. If $w$ is the issuer of the received assertion (step 3), $v$ sends a request to revoke the link to $w$ to avoid link inconsistency (step 4).

Otherwise, $v$ sends a data update request to $w$ (step 6). If $v$ has no need to update its original data using the received data (step 8), it does not create a new link to avoid cyclic links and cancels the data transfer (step 9). If the condition of step 1 is not true (step 11), a new link is created (step 12), its transfer identifier and parent information are set (steps 13 and 14), incoming assertion $a^{u,v}$ is registered to the new link (step 15), and the link is added to link table $L(v)$ (step 16). The procedure from step 12 to 16 corresponds to that for an entity that receives a request from a DM entity that processes Alg. 1.

### 4.5 Data Updating

A DM entity that transfers an assertion to a DC entity and keeps its corresponding transfer link needs to update the link when the data contained in the assertion is updated. Algorithm 6 is used by DM entity $p$ to update the data and manage the link after the data is sent by means of assertion $a^{\rho,q}_0$ to DC entity $q$.

In Alg. 6, a new signed assertion is created that inherits the transfer identifier and policy of previous assertion $a^{\rho,q}_0$ (step 1 to 5). Assertion $a^{\rho,q}_0$ is deleted from the set of outgoing assertions in the link table (step 6), new assertion $a^{\rho,q}$ is registered in the table, and the link itself is kept (step 7). Finally, $a^{\rho,q}$ is sent to $q$.

Algorithm 7 is used by DC entity $q$ when it receives a
Algorithm 6 update_data(p, q, a_{p,q}^q)

Require: ∃n ∈ N; l_n ∈ L(p), l_n.trans_id = a_{p,q}^p.trans_id, a_{p,q}^q ∈ l_n.OA
1: a_{p,q}^q ← create_assertion(a_{p,q}^q)
2: a_{p,q}^qIssuer ← p
3: a_{p,q}^q.trans_id ← a_{p,q}^p.trans_id
4: a_{p,q}^q.policy ← create_policy(p, q, a_{p,q}^p.policy)
5: sign(a_{p,q}^q)
6: l_n.OA.del(a_{p,q}^q)
7: l_n.OA.add(a_{p,q}^q)
8: send_assertion(q, a_{p,q}^q)

In Alg. 7, r is made a child of q if there is existing link l_m for incoming assertion a_{p,q}^p (step 1). For every outgoing assertion a_{i,j}^r, new assertion b_{i,j}^r is created using updated assertion a_{p,q}^q (step 3), and the data update operation is executed using b_{i,j}^r for Alg. 6 (step 4).

5. TRANSFER BROKERING

As mentioned in Sec. 4.1, it may be inappropriate for an entity that receives an assertion as a DC to then forward it to other entities as a DM, for reasons of privacy or policy. A good solution in such cases is for the entity (broker) to ask other possible DM entities to issue a new private assertion to a possible DC entity if it is needed or at the DC entity’s request. In other words, the broker entity mediates the creation of a new transfer link.

Figure 6 illustrates an example of transfer brokering. Entity q, the brokering entity, does not create a transfer link for entity r; instead, entity p (the parent of q), creates a new link.

(a) r requests data from q. (b) q mediates link creation between p and r.

Let us consider the use case of transfer brokering for delegated access [4]. Figure 7 depicts the interactions among a DO, an identity provider (IdP), a service provider (SP), and a resource. This approach extends the security assertion markup language (SAML) Web single sign-on and artifact resolution profiles [9], but its basic idea does not depend on the specifications. Steps 1 to 7 are the same as the interactions in the SAML profiles, which are shown in the figure as an introduction to the extension presented here.
6. INFORMATION PRACTICES TRACING

A tracing method is used to determine how identity data is handled and the management policy is enforced.

6.1 Data Usage Report Protocol

The tracing function of a DM entity instructs each DC entity to provide a report of how data was used and how the related policies were enforced after data received from the DM has been used. By means of this report, each DC entity can demonstrate that it has properly enforced the agreed upon policy. As the reports from a DC entity become more positive, the entity earns more trust and enhances its reputation.

Figure 8 shows an example of data usage report request message in XML description.

Figure 8: Data Usage Report Request.

The requested information includes the <DataManagerID>, <TransferID>, and <DataConsumerID>, which are the DM entity identifier, the transfer identifier, and the DC entity identifier, respectively. <ReportURL> indicates the URL where the recipient should enter the response. <Duration> is the time interval during which the corresponding events take place. <Attributes> are attribute information that are to be reported. The attribute of request element Recursive indicates that the requestor directs the receiver to perform another recursive reporting request if certain conditions are satisfied, which is explained in Sec. 6.2.

Figure 9 shows a sample response to the request in Fig. 8.

The response contains a report showing the specified attribute was used at a particular time instant by the DC entity after the entity received an assertion containing the attribute information. In this case, the entity propagated attribute BMI information to entity sp2.com.

This protocol optionally requests the current status of identity data for the purpose of a monitoring test. For instance, a DM entity requests the value of an attribute although it knows the agreed upon policy does not allow retention of that value. If the DC entity reports a value, it has retained the data inappropriately. The DM can thus deduce that the DC entity has not complied with the data management policy.

6.2 Recursive Tracing

When a DM entity receives a usage report from a DC entity and if the DM entity plays a DC role relative to its parent entity, the DM entity recursively reports to its parent in the direction opposite that of the arrow in the data transfer link (reporting request chaining). Initially, there are successive data usage report requests in the direction of the transfer links, as shown in Fig. 10.

<table>
<thead>
<tr>
<th>(1) Data Usage Report Request</th>
<th>(2) Data Usage Report Request</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
</tbody>
</table>

Figure 10: Usage Report Request Chaining.

DM entity e₀ initially sends a data usage report request to DC entity e₁. Entity e₁, acting as a DM, then sends a request to entity e₂. The series of responses corresponding to the requests are shown in Fig. 11.

<table>
<thead>
<tr>
<th>(3) Data Usage Report Response</th>
<th>(4) Data Usage Report Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Diagram" /></td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
</tbody>
</table>

Figure 11: Usage Report Response Chaining.

DC entity e₂ sends its data usage report to e₁. Entity e₁ creates its report by integrating data from the log produced by its data management reporting function with the data received from e₂ and sends the report to e₀.

By checking the report from e₁, DM entity e₀ can ascertain whether both DCs have operated in conformity with the policy to which both agreed prior to the attribute exchange. If it detects inappropriate operation, it can request improvement in the DC entity’s policy enforcement or it can update the trust evaluation value for the DC entity in the trust management function.

7. POLICY BINDING AND CONSTRAINTS

Let us consider an example privacy policy binding mechanism and an agreed upon policy description.

Figure 12 shows an example policy embedded assertion extended using SAML [8]. For improved readability, prefix descriptions are omitted in the XML schema. Since SAML...
Figure 12: Policy Embedded Assertion.

As Fig. 12 shows, element <Advice> contains data transfer and policy information. <TransferID> denotes the identifier assigned to the assertion assertion propagated by the issuer (idp.com) to the recipient (hospital1). <Policies> contains the agreed upon policy between the issuer and the recipient that describes the privacy constraints in a manner similar to that used in P3P specifications [13].

The agreed upon policy specifies that the attributes categorized into attribute groups personalData and medicalHistory can be used for the purpose of “surgery” by the recipient hospital1 and stored for eight years. Over-rule indicates that the preference of the DO (Subject) may overrule the agreed upon policy if the DO needs to make a decision or consent to some action of a DC entity.

Element <Tracing> is a characteristic of the proposed model and presents the recipient with the tracing and reporting requirements. The Recursive attribute means that the recipient is required to send a recursive request to a child for the received tracing report, which is explained in Sec. 6.2. Element <ReportURL> indicates the URL to which the data usage reports are to be entered. The Directive attribute indicates two types of values for using the data usage report protocol. The Report value denotes a simple report whereas the Test value indicates that the recipient will receive a monitoring test message from its DM.

8. DISCUSSION

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

8.1 Trust

The proposed model assumes that trusted entities manage, exchange, use, and trace identity data in a collaborative fashion. Since untrusted entities may break an agreed upon policy and use identity data in an unauthorized manner, trust management is an essential issue.

The proposed framework is effective for managing trust because it provides a means for both a participating entity and a user to manage and update the trustworthiness of entities if trustworthy reports are accumulated from the same entity. In the model, trust entities can strengthen a trust relationship by making detailed reports periodically. The DO’s feedback on the reports affects the trust evaluation of entities.

8.2 Traceability

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

As illustrated in the scenario described in Sec. 1, privacy constraints have a variety of aspects. Therefore, in existing identity management systems, it is difficult for DOs to determine the status of their data after the data has been propagated, especially when the data has been propagated from entity to entity. The proposed framework supports the determination of the current status and the tracking of the usage history for identity information by the DOs.

8.3 Accountability

Entities are at risk of involvement in inappropriate data handling or disclosure if some 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 policy. The tracing and reporting schemes used in the proposed model enable entities to improve accountability for their identity practices. The exact and detailed reports produced help them to earn more trust from other entities.

Auditing is an alternative method for entities to demonstrate the correctness of their identity data practices. An auditing authority monitors the identity management log for legal or compliance reasons. In contrast, the proposed model takes a decentralized approach – a DO can ask a DM entity to aggregate the data usage reports of its children in order to get a clear understanding of the status and usage of the DO’s identity data.

8.4 Policy Enforcement

Identity 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. In the proposed model, if data leakage or misuse takes place, the tracing results and usage reports can be used to investigate and identify the cause of a leakage or misuse event and to determine where responsibility for it lies.

Although the policies in the proposed model are typically...
privacy-related ones, security-related ones (e.g., authorization in Sec. 3.1) could be used as well. If the data is personal attribute information, the owner’s preferences generally have priority over the DM’s policy. In contrast, if the data is related to the DO but is not owned by the DO, the DM’s policy may have priority. For example, although the information of a privilege for enjoying particular digital content is related to a DO, the administrator may wish to enforce the policy that only the DO can exercise the privilege to prevent the information’s unauthorized distribution for an administrative reason.

8.5 Identity Tracking

The initial DM entity may aggregate data usage reports from several DC entities. Since a data usage report includes every action a DO takes related to identity data, this data collection may lead the DM entity to accumulate private information about the DO. In the proposed model, a transfer identifier is associated with every data transfer containing identity data. The identifier is a dynamic and transient one that covers limited actions related to the usage of identical identity information.

There is thus a trade-off between the risk of private information leakage and convenient aggregation of usage reports. In the proposed model, a DO can select which entities to trust and decide on the degree to which his/her data usage events are traced.

To avoid tracking every event related to a DO, a user-agent of the DO could aggregate the data usage reports by means of a pseudo transfer identifier, thereby preventing other entities from associating transfers with a DO. Or data usage reports could be limited to such information as the URL to access the report at the corresponding entity. These are also open issues.

8.6 Identity Distribution

The proposed framework promotes identity data distribution among entities in a privacy preserving manner. There is a trade-off between seamless identity distribution among multiple entities and adequate privacy protection between a pair of entities since the propagation of a private assertion to an unauthorized entity would be nothing else but the intentional divulgation of private information, as explained in Sec. 4.3.

Therefore, the distribution of an assertion by transfer chaining may not be an appropriate identity transfer pattern in some cases from the viewpoint of privacy protection. In such a case, transfer brokering may be more appropriate. The proposed framework enables a DO to select the best pattern from both usability and privacy viewpoints in a user-centric way.

9. CONCLUSION AND FUTURE WORK

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

Future work includes enhancement of the model to ensure compliance with the policies upon which entities have agreed and to enable integration of strong auditing information from a trusted third party.

10. REFERENCES


