An attribute assurance framework to define and match trust in identity attributes

Ivonne Thomas
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
ivonne.thomas@hpi.uni-potsdam.de

Christoph Meinel
Hasso-Plattner-Institute
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
meinel@hpi.uni-potsdam.de

***Abstract*—Identity federation denotes a concept for the controlled sharing of user authentication and user attributes between independent trust domains. Using WS-Federation, service providers and identity providers can set up a Circle of Trust, a so called federation, in which each member is willing to trust on assertions made by another partner. However, if a member has to rely on information received from a foreign source, the need for assurance that the information is correct is a natural requirement prior to using it. Identity assurance frameworks exist that can be used to assess the trustworthiness of identity providers. The result of this assessment is a level of trust, that can be assigned to an identity provider. However, existing approaches for evaluating identity assurance do not allow to define trust levels for individual attributes.**

In our trust model, we consider both: (a) trust in an identity provider as the issuer of assertions and (b) trust in single attributes that an identity provider manages. In this paper, we show how our approach that we implemented in a logic-based framework can be used in web service scenarios to provide trust information on the level of identity attributes, especially about the verification process, and to match trust requirements of attributes during request processing.

***Keywords*-Identity and Attribute Assurance, Identity Federation, Trust**

I. Introduction

The open and decentralized nature of Service-oriented Architectures demands for new security concepts that take these characteristics into account. In the field of identity management, open identity management models ([1], [2]) such as the decentralized or federated identity management emerged to specifically address the needs of sharing identity information across multiple security domains. Designated entities to offer identity services, called *identity providers (IPs)*, are at the heart of these new models. Identity providers perform the task of authenticating users as well as managing their identity data. Service providers can request identity data from an identity provider and use it for example in their access control decisions.

A necessity to share identity information across security domains is the willingness of involved parties to trust on information that is received from a foreign domain. In order to establish this trust, a relying party ought to know how the information received from an identity provider has been gathered, stored and processed inside the organization offering the identity service. In fact, the whole process, technologies, protections, infrastructure and other safeguards in place, on which assertions are based influence the confidence a relying party such as a service provider can put in received assertions - reaching from organizational aspects such as the legal situation to technical details such as the used token type to transfer identity information from the identity service to a requesting party.

Identity assurance frameworks have been developed by initiatives and organizations around the world with the goal to align different trust requirements by defining a common, standardized set of trust criteria to assess an identity provider and its systems. Often these trust criteria are clustered and assigned a level of trust or in case of digital identities a level of assurance (LoA). A level of trust or level of assurance (LoA) reflects the degree of confidence that someone can assign to the assertions made by another party, such as an identity provider, with respect to the users identity information.

Using current identity assurance frameworks, identity providers are assigned a level of trust according to the criteria they fulfill. However, looking at the online world, we find very varying trust requirements to use a service. Often it does not matter whether our online identity matches with our real-life identity as long as it always the same identity we use to interact with a web site or service. On the other side, as soon as a risk is involved in transactions, a service will require the assurance that an online user matches a real-life identity in order to held him/ her liable in case anything bad happens. This distinction between verified identities and identities containing self-asserted attributes is not in the scope of current assurance frameworks. Also, using existing assurance frameworks, it is hard to reflect possible changes of a user's identity trust level over time. As identity proofing processes are cost-intensive and time-consuming due to the effort required to verify identity attributes, a verification of an attribute might not be desired as long as a user is not involved in transactions that require it. Therefore a user might decide to register with an identity provider without proper identity proofing, having for example his/ her name self-asserted and getting involved in the identity proofing process only upon concrete demand. This requires a different trust level per user and does not allow to rate an identity provider as a whole.

In Thomas et al. [3], we argued that in order to reflect these differences, it is not sufficient to assign a single trust level to an identity provider. Instead, additional trust levels for attributes are needed in order to reflect the assurance quality of different digital identities hold by an identity provider. In [4], we propose a two layered trust model that considers both: the overall trust into an identity provider and the trust into the identity of a user.

In this paper, we present our framework based on our two-layered trust model, that allows to represent trust in an identity provider as well as trust in the identity attributes. We further defined general verification classes as a possible way to describe the verification of attributes in our model. In our framework we use an approach based on Horn logic, which allows us to easily reason over considered entities and its trust characteristics. In this paper, we present this framework and give two use cases to show its applicability in different scenarios. In particular, at the identity provider side, we match requests for attributes with a particular verification need with the available attributes in the provider. At the service provider side, we use our framework during policy creation to find suitable identity providers that match the given trust requirements.

The rest of paper is structured as follows. The following section describes a small motivating scenario that highlights our vision, that is addressed by our research. Afterwards, Section II considers related work in the field of identity assurance. Section III presents our attribute assurance framework including a formalization of our model and details about its implementation. Section IV describes two use cases, in which we applied our framework. Finally, Section V concludes the paper.

A. *Motivating Scenario*

Our vision is an open world, in which identity providers are commonplace and users can choose freely between them to manage their identity information and to issue it to service providers that require this information. As service providers can have very varying requirements for identity attributes and their quality, users are facing the task to choose among many identity providers the one(s) that match with the service providers needs.

In our vision, which is illustrated in Figure 1, each identity provider holds a set of attributes with possibly different qualities of trust. A bank could for example reliably assert a user's payment details while a university could assert whether a user is a student. Also, with regard to the recent launch of electronic id cards in various countries, an identity provider, which has been approved to access the users'

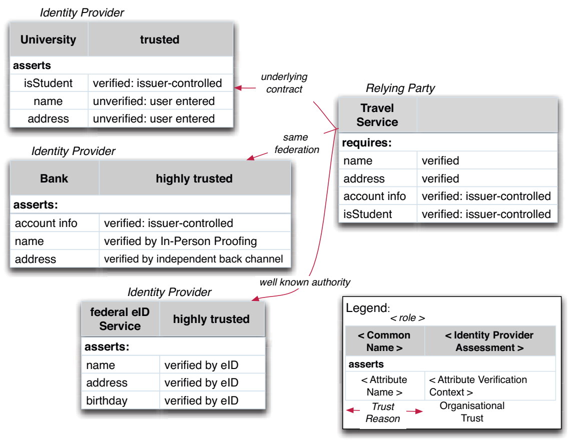

Figure 1. Vision of a scenario with multiple identity providers issuing identity attributes with different qualities of trust.

electronic id cards can assert reliably attributes as name, address or date of birth.

As opposed to the identity providers, there are service providers that need certain identity attributes of a user and take the role of the relying party. Depending on the intended transaction, the service provider might require different degrees of assurance of a user's identity and its attributes.

Current technologies as OpenID [5] or Information Card [6] only offer the possibility to express (a) which attributes are required and (b) whether any identity provider or a specific one should be used. Using Web Service technologies possibilities are a bit broader: WS-SecurityPolicy [7] allows to state a list of required attributes *per* issuer (identity provider).

In our scenario, we aim at expressing not only which attributes are required and which identity provider(s) are suitable to assert them, but also which trust level an issued attribute value should have as well as which properties should apply for the issuing identity provider. To give an example, we want to formulate requirements such as "The relying party requires an attribute *name* from the user who proved his name by registering *in-person* at a *federated* identity provider.".

II. Related Work

The problem of identity assurance has been addressed by various initiatives and organizations around the world. At first driven by governments in various countries (cf. for example UK Office of the e-Envoy [8], NISTs Special Publication 800-63 "Electronic Authentication Guideline" [9], etc.), in recent days also industry and educational institutions joined the effort of defining identity assurance guidelines with the goal to provide a more trusted interaction between online partners. As a result a number of so called identity assurance frameworks have been designed with the one provided by the InCommon federation ([10], [11]) being the most comprehensive approach among them. As said before, identity assurance frameworks usually define a number of trust levels that can be assigned to an identity provider. However, trust levels for attributes are not in the scope of current identity assurance frameworks.

Work regarding trust levels for attributes has been conducted by Chadwick et al. in [12]. Chadwick et al. build on NIST's [13] concept of assurance levels. Similar to our work, they propose to have separate metrics for identity proofing processes (expressed in the Registration LOA) and the authentication of a subject (expressed in the Authentication LOA). Authentication LOA and Registration LOA are combined to a Session LOA and send in each assertion from an identity provider to a service provider. Compared to this, in our work, we assign trust levels not only to the registration and authentication process, but to individual attributes. For this purpose we define verification classes in a hierarchical manner to express details on the verification of attributes at an identity provider. Our motivation is to provide more choices for a relying party's policy to express its demands and for identity providers to express their offers.

Mohan et al. [14] provide a framework, called AttributeTrust, for evaluating trust in aggregated attributes. Attributes are provided by trusted attribute providers. Trust in these attribute providers in turn is calculated by using a reputation system model. In their approach, service provider express their confidence in other entities to supply trusted attributes. After each successfully completed transaction, service providers are asked to provide feedback for attribute providers. Over time, chains of confidence are formed between service providers and attribute providers. Compared to our approach, AttributeTrust builds up a global graph by aggregating feedback of multiple parties to express the confidence in other entities to assert trusted attributes. For this reason, as opposed to our work, individual assessments, based for example on the fact that a relying party has a contractual relationship to an attribute provider, are hard to express using the AttributeTrust framework. Also, Mohan et al. do not differentiate between trust in attributes and trust in the attribute provider itself.

In her PhD thesis, Bharghav-Spantzel[15] establishes the notion of two types of assurance, namely validity assurance and ownership assurance. Validity assurance refers to the correctness of information while ownership assurance refers to the confidence that a claim actually belongs to a Subject. Her work is part of the VeryIDX[16] identity management system that fosters anonymisation techniques such as the *Zero-knowledge proof of knowledge* protocol to preserve the users' privacy. Assurance levels are used to describe the ownership and consistency of identity records. Compared to our approach, Bharghav-Spantzel also suggest to consider trust aspects of attributes more diverse and to have different levels for the correctness of information and for the subject-to-account mapping. In our research, we go a step further and describe attribute assurance details on an even more detailed level. To give an example, our assurance framework allows not only to define different trust levels, but also to express certain properties as part of our assurance model such as the used verification of an attribute or whether an identity provider is part of the same federation.

Baldwin et al. present an identity assurance framework in [18], which clusters assurance aspects for identity trust. The approach classifies aspects according to the identity management lifecycle consisting of a registration phase, the maintenance phase and a destruction step when the digital identity is not needed any longer. As in most identity assurance frameworks trust in attributes is only considered during the registration phase.

III. Attribute Assurance Framework

This section presents our attribute assurance framework which is based on our two-layered trust model, which has been presented in earlier work [4].

A. Organizational and Identity Trust

In our two-layered model, we distinguish between two types of trust. First, we consider the trust in the identity provider's operator, basically its readiness to support and operate a reliable operational service. This includes the legal situation, laws and guidelines an identity provider adheres to or the storage of user data on its internal systems. In our model, we call this aspect *organizational trust*. Organizational trust manifests often in federation agreements and contracts between the involved parties.

Identity trust on the other hand as stated in [4] is mainly characterized by factors that are subject to vary between different digital identities of the same or different users within an identity provider. Such factors are for example the authentication process and the subject-to-account mapping, as well as the trustworthiness of the subject's attributes and the token.

B. Formalization of our Trust Model

Our attribute assurance model consists of certain concepts such as identity providers $\mathcal{I}$, attribute types $\mathcal{T}$, verification classes $\mathcal{V}$, etc. as well as relations among them and properties that apply in particular to the concept of identity providers as for example $\mathit{isFederationPartner}(i) = \mathit{true}$ for $i \in \mathcal{I}$. In particular, let

- $\mathcal{S}$ be a set of service providers,
- $\mathcal{I}$ be a set of identity providers,
- $\mathcal{P}$ be a set of participants with $S, I \subseteq P$,
- $\mathcal{O}$ be a set of organizational trust levels (identity provider trust levels),
- $\mathcal{A}$ be a set of attributes,
- $\mathcal{T}$ be a set of attribute types and
- $\mathcal{V}$ be a set of attribute verification classes

Our model comprises the following relations:

1. Concerning attributes, attribute types and identity providers:

- First of all, identity providers are able to assert certain user attributes of a certain attribute type $t \in \mathcal{T}$, that is *Assert* $\subseteq (\mathcal{I} \times \mathcal{T})$.
- Each attribute is assigned a type: $h : \mathcal{A} \mapsto \mathcal{T}$
- *Applies* $\subseteq (\mathcal{V} \times \mathcal{T})$ denotes that an attribute verification class is applicable to an attribute type.
- $AttrTrust : (\mathcal{I} \times \mathcal{T}) \mapsto 2^{\mathcal{V}}$ whereas $(\mathcal{I} \times \mathcal{T}) \in Assert$ maps attributes hold by an identity provider to a set of verification classes that are supported by the identity provider.

2. Concerning attribute trust and attribute verification contexts

- Each attribute verification context is represented by an attribute verification class $v \in \mathcal{V}$.
- Attribute verification classes can be put in a hierarchical structure. $subClassOf : \mathcal{V} \mapsto \mathcal{V}$ denotes that $v_1 \in \mathcal{V}$ is a sub class of attribute verification class $v_2 \in \mathcal{V}$: $subClassOf(v_1) = v_2$

3. Concerning organizational trust and identity providers properties.

- Each identity provider $\mathcal{I}$ is characterized by a set of properties $\mathcal{P}$.
- A property $\mathcal{P}$ is a triple $\langle n, pr, v \rangle$, whereas $n \in N$ is a property name, $pr \in Pr_n$ is a binary predicate and $v \in V_n$ a value from the set of possible values for n.
- The following predicate symbols are considered: $Pr = \{=, \neq, <, >, \geq, \leq\}$. Pr_n is the set of predicates that is applicable to a given $n \in N$.
- $OrgTrust \in N$ is a special property describing the overall assessment of the trust relationship of a relying party to an identity provider. $V_{OrgTrust} = \mathcal{O}$ is the set of possible values.
- *hasProperty* $\subseteq (\mathcal{P} \times \mathcal{I})$ maps a property to an identity provider.
- Given the special property $\langle OrgTrust, Pr, O \rangle \in \mathcal{P}$, $(\langle OrgTrust, Pr, O \rangle \times \mathcal{I}) \subseteq hasProperty$ denotes that an identity provider is assigned an organizational trust level.

Facts, rules and queries: All assurance knowledge is presented as facts and rules, that are based on predicate logic, in particular on Horn clauses. Facts are definite Horn clauses consisting of exactly one positive literal, therefore exactly one predicate p. Rules are Horn clauses consisting of a disjunction of one positive and at least one negative literal. They are usually written as implications:

$$p_1 \leftarrow p_2 \wedge \cdots \wedge p_n.$$

Predicates are all of the above defined relations and properties, e.g. $Assert(i_1, t_1)$ evaluates to *true* if $i_1 \in \mathcal{I}$ asserts attributes of type $t_1 \in \mathcal{T}$.

Facts and rules form together the *knowledge base*.

- A knowledge base $\mathcal{K}$ contains a set of facts that is trusted by the ones relying on this knowledge base: $\mathcal{K} \subseteq \mathcal{F}$. A knowledge base can be shared by multiple participants, that is $K_{e_1,..,e_n}$, $e_1, .., e_n \in \mathcal{E}$, or belong to a single entity, such as K_r for a relying party $r \in \mathcal{R}$.

Queries can be formulated in order to reason over the knowledge base. A query is a conjunction of positive literals. It has the form

$$p_1 \wedge p_2 \wedge \cdots \wedge p_n.$$

Example: Consider a relying party $r \in \mathcal{R}$ that is in the same federation as the identity provider i_1. As an example, the relying party could have the following facts in its knowledge base $\mathcal{K}_r$.

First, the relying party knows that identity provider $i_1 \in \mathcal{I}$ can assert attributes of type $t_1, t_2 \in \mathcal{T}$ Therefore,

$$Assert = \{< i_1, t_1 >, < i_1, t_2 >\} \in \mathcal{K}_r.$$

Furthermore, $V = \{v_1 = \text{"verified"}, v_2 = \text{"certificate-based"}, v_3 = \text{"unverified"}\}$ denote available attribute verification classes to describe the attribute verification whereas v_2 denotes a concrete verification mechanism of v_1: $subClassOf(v_2) = v_1$. We assume, that the attribute verification class $v_2 \in \mathcal{V}$ is applicable to attribute type t_1. t_1 could for example be the email address of a user, which is provable by a an email certificate. We further assume that the attribute verification classes v_1 and v_3 are applicable to all attribute types. Therefore,

$$Applies = \{< v_2, t_1 >, < v_1, t_1 >, < v_1, t_2 >, < v_3, t_1 >, < v_3, t_2 >\} \in \mathcal{K}_r.$$

In our example, we assume that the identity provider i_1 can verify attributes of type t_1 using the attribute verification class $v_2 = \text{"certificate-based"}$. Also the other classes v_1 and v_3 are supported. Therefore

$$AttrTrust(i_1, t_1) = \{v_1, v_2, v_3\}.$$

Attributes of the second attribute type t_2 are always unverified when managed by i_1. Consequently,

$$AttrTrust(i_1, t_2) = \{v_3\}.$$

In order to characterize the organizational trust relationship, the following properties are considered by the relying party. First, the relying party considers whether an identity provider is in the same federation as itself and second, the relying party considers the ICAM trust level an identity provider is certified with. We have $N = \{n_1 = OrgTrust, n_2 = FederationPartner\}$ as the possible property names with $Pr_{OrgTrust} = Pr$ and $Pr_{FederationPartner} = \{=, \neq\} \subset Pr$.
$V_{OrgTrust} = \mathcal{O} = \{\text{"ICAM Level 1"}, \text{"ICAM Level 2"}, \text{"none"}\}$ are the available organizational trust levels and

$V_{\mathit{FederationPartner}} = \{true|false\}$ are the possible values for n_2.

We said, that the identity provider i_1 is part of a federation, therefore

$$\mathit{FederationPartner}(i_1) = true.$$

Furthermore, i_1 is certified with an ICAM trust level of 2, therefore

$$\mathit{OrgTrust}(i_1) = \mathit{ICAM\ Level\ 2}.$$

C. Implementation

We implemented our model described in the previous section III-B as a Java-Prolog based framework that can be included in various implementations of identity providers, web service clients as well as web services. In summary, our attribute assurance framework allows to

- define a global knowledge base containing supported attribute types, supported verification classes per attribute type as well as known identity providers, their properties and trust levels. Furthermore, verification classes of attributes can be defined in a hierarchical order.
- assign verification classes to concrete attributes of users
- query the knowledge base for available identity providers that can assert a given set of attributes and verification requirements
- match a request for an attribute with a certain verification class to the available attributes

We choose to use Prolog as a programming language as it provides a declarative way of presenting our knowledge base. Each Prolog program is made of facts and rules that operate on the set of facts. We can easily add new facts and define additional rules to match the requirements of a given scenario. Furthermore Prolog allows us to reason over the given set of facts while at the same time providing us the necessary flexibility in defining these facts. As each web service scenario can demand for different verification classes and properties that one considers to assess an identity provider, a framework to add assurance information to attributes should be able to deal with this flexibility.

The core of our framework is the knowledge base consisting of a set of facts, such as "Identity provider i_1 can assert attribute a_1 with a trust level v_2." and rules such as "All identity provider within the federation are trusted".

We provide an API to configure the knowledge base according to the model. Once defined, multiple kinds of queries can be formulated to the knowledge base. The following list gives some examples:

- Find all attributes $a \in \mathcal{A}$ an identity provider $i \in \mathcal{I}$ can assert with a trust level of $v \in \mathcal{V}$ and $v =$ *"verified"*
- Find all identity provider $i \in \mathcal{I}$ that are in a given federation and can assert a verified attribute $a =$ *name* that has been proven by in-person registration at the identity provider ($v =$ *in-person*).
- Find all identity providers $i \in \mathcal{I}$ with a trust level of *at least 3* according to the ICAM [19] specification of identity assurance levels that can provide a verified student attestation $a =$ *isStudent* with $v =$ *verified.*
- Find a set of identity providers that can provide a verified email address issued by *any* identity provider and the *name* and *age* of the user issued by an *authorized eID Service* (electronic ID card).

IV. Use Cases

In this section, we show how our attribute assurance framework described in the previous section III can be used in web and web service based scenarios to augment open identity management solutions with the ability to deal with attributes of different trust qualities. In the first use case, we focus on the identity provider side and show how we can use our attribute assurance framework to match an incoming request for a verified attribute with the attributes which are available at the identity management system of the identity provider.

A. Matching verified attributes at an identity provider

The core of this use case is our implementation of a trust-aware identity provider, which is shown in Figure 2. It features general identity management system functionality including creating, editing and deleting multiple digital identities and associated attributes, the option to verify certain attributes as well as interfaces to request, issue and sign security tokens. Supported and used standards are WS-Trust 1.3 [20], SAML 1.1, SAML 2.0 [21] as well as Information Card [6] and the OpenID 2.0 Authentication specification [5].

Our identity provider, whose architecture is shown in Figure 2, is used by students of the university to authenticate with various web as well as web service based relying parties in and outside the university. Students have the choice to go through a verification of certain attributes whereas several verification methods are supported. For an email address for example, a user can provide an email certificate, which proves that s/he has the entered email address or an email including a confirmation link can be send to the user. As each student is also issued an email address from the university, this email address is verified by default as the university as the issuer can reliably assert that the email address exists and belongs to the user. We define different categories of verification, called verification classes, for each kind of verification. In our email example, the corresponding verification classes are *proofByCertificate*, *backChannelProofing* and *issuer-controlled*.

1) Defining the knowledge base: In order to define the knowledge base, we need to define all supported attributes, a hierarchy of verification classes, and assign attributes to verification classes. As said before, we defined a number of

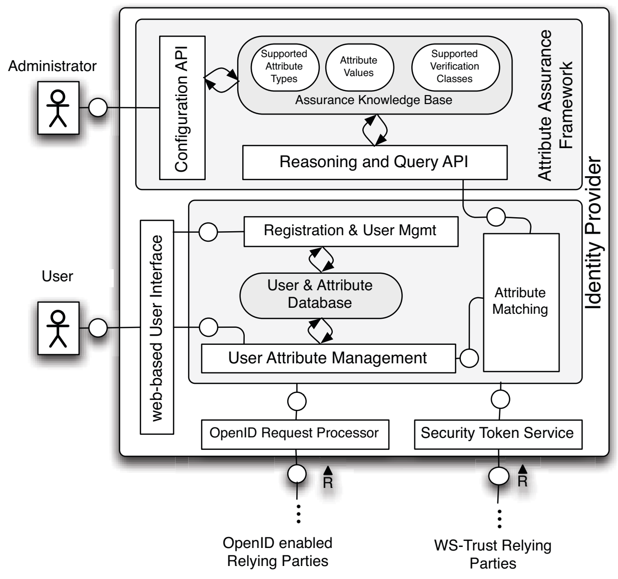

Figure 2. Architecture of our identity provider using the Attribute Assurance Framework.

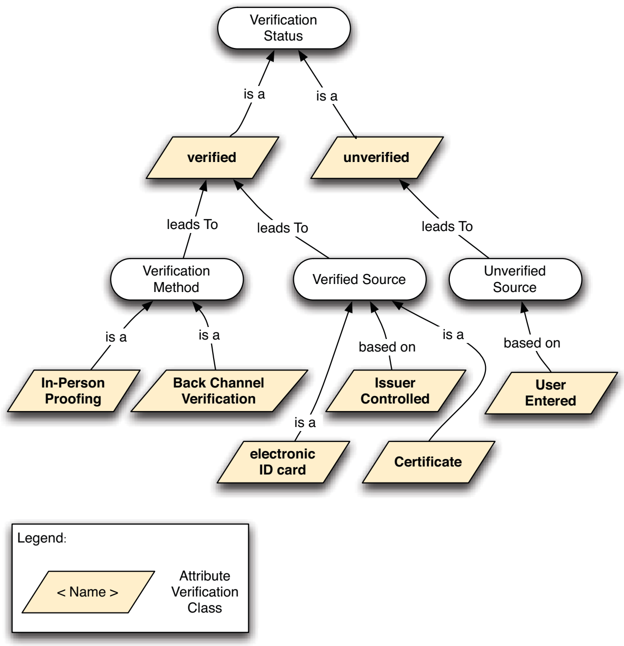

Figure 3. Defined Verification Context Classes.

verification classes for our purposes that can be used in various scenarios to specify attribute verification requirements. When defining these classes for our use case, we tried to find general verification schemes that can be applied to several attributes, but might be implemented in different ways. For example, the verification of an attribute by an independent back channel can be done for email addresses by sending an email to the claimed address with a verification link in it. The same scheme can also be used to verify a bank account. In this case a small amount of money (1 cent) is usually transferred to the claimed account with a password in the transaction data, that the user needs to enter later on to prove that s/he is the owner of the account. Altogether, we defined six verification classes plus the two all-comprising classes *verified* and *unverified*, which are shown in Figure 3.

Besides the verification classes, our knowledge base hold a number of supported attributes. In our implementation, we support the list of well-known claim types that has been defined by Microsoft in [22]. Each attribute corresponds to a claim type URL as defined by the InformationCard Interoperability specification [23] and to an OpenID Attribute Type as defined by the OpenID Attribute Exchange specification [24].

2) Used Claimtypes: We mapped our verification classes to current technologies by defining custom claim types. Each attribute is assigned a number of URIs that identify the attribute together with the used verification in a global context. In our implementation, we use the following name scheme to identify a verified attribute:

- for InformationCard:
  `http://openid.hpi.uni-potsdam.de/icschema/<Attributename>/<VerificationContextClassName>`
- and for OpenID:
  `http://openid.hpi.uni-potsdam.de/axschema/<Attributename>/<VerificationContextClassName>`

3) Querying the knowledge base: When a request is received from a relying party for a certain attribute either at the *Security Token Service (STS)* endpoint or at the *OpenID Request Processor*, the *User Attribute Management* as shown in Figure 2 is queried to retrieve the attribute values of the Subject of the request from the *User & Attribute Database*. Afterwards, the *Attribute Matching* component checks whether the requested verification class matches with the verification context of the attribute of the user. As verification classes can be hierarchical, a request for a verified email address could result in an email attribute with one of the context classes *issuer-controlled*, *backChannelProofing* or *proofByCertificate*. If the match succeeds, the attribute value is returned. Otherwise a message is returned to the requestor.

B. Generating policies based on assurance information

In the second use case, we use our attribute assurance framework to find those identity provider(s) among a group of available identity providers that matches with a service providers requirements for verified attributes. Looking at current web service scenarios, a service provider usually specifies a list of trusted identity providers to retrieve attributes from. Using our framework, we foster a scenario, in which identity providers are holding different sets of users' identity attributes with possibly different trust qualities. Companies, for example, might provide security token services for their employees to collaborate more effectively with partners by creating or joining federations between their own and the foreign security token services. Such a security token service associated with an organization can keep track of its users in a much better way than identity providers for the web, which do not restrict their service to a specific group of people. In fact, with the latter ones often the registration of users takes place online without any verification. As both kinds of identity providers are beneficial depending on the intended use, our framework can be used to focus on these differences by stating for each identity provider not only whether it is trusted or not, but also which of the attributes are verified. With this knowledge a service provider can specify its trust requirements per attribute and find a combination of suitable identity providers.

Our use case consists of a service provider offering an *Order Service*, which is used by a client included in an online store to order items. As can be seen in Figure 4, each request to the *Order Service* is intercepted by a policy enforcement point. An access request is send to the *Policy Decision Point*, which checks based on the policy whether the *Subject* of the request is authorized to access the service. In order to define the policy, a service's requirements for attributes are matched with the *Assurance Knowledge Base*.

1) Defining the knowledge base: In the knowledge base, we define

- a number of available identity providers, e.g. $I = \{i_1 = \textit{"University IP"}, i_2 = \textit{"Bank IP"}\}$
- the attributes, they can assert, e.g. $\textit{Assert} = \{< i_1, \textit{isStudent} >, < i_1, \textit{name} >, ..., < i_2, \textit{name} >, ...\}$.
- the corresponding verification classes, e.g. $\textit{AttrTrust}(i_1, \textit{name}) = \textit{"user-entered"}$, $\textit{AttrTrust}(i_2, \textit{name}) = \textit{"In-Person"}$
- the organizational trust levels, e.g. $O = \{\textit{"trusted"}, \textit{"highly trusted"}\}$
- further rules and properties, such as whether an identity provider is federated or that all federated identity providers have the organizational trust level *trusted*.

In our scenario, the knowledge base is defined by the service provider's IT security management and configured by an administrator.

2) Defining trust requirements of the service and generating policies: For each service of the service provider, a policy is defined and enforced. Such a policy depends on the risk involved with a transaction. For the *OrderService*, we

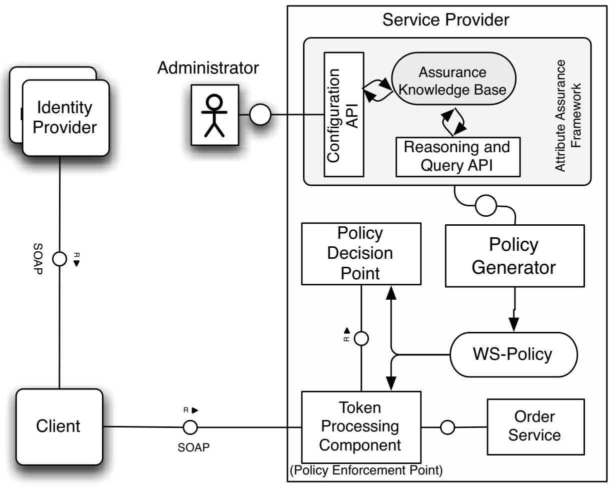

Figure 4. Architecture of the service provider using the Attribute Assurance Framework.

could for example assume, that an attribute name is required from the user which registered in-person and showed a valid ID card. The identity provider $i \in I$ needs to be a federated IP:

$$(\textit{isFederated}(i), \textit{Assert}(i, \textit{name}), \textit{AttrTrust}(i, \textit{In-Person}))$$

Given the request, a reasoning is done over the *Assurance Knowledge Base*. The result is a list of possible identity providers that can be used in the policy or an empty list in case the request can not be matched. If more than one combination is possible, either all combinations are written down in the policy file, which however requires a client that can deal with multiple policy alternatives or one is chosen by the *Policy Generator*. The resulting policy file is compliant to the standards WS-Policy [25] and WS-Security Policy [7] and can be used with any standards-based policy enforcement point.

Another situation that can occur when several attributes are required by the service, is the need to aggregate attributes from different identity providers. In our scenario, this is done by the client. When the client receives the policy from the service and the policy demands security tokens from different identity providers, the client will query them one by one, aggregate them and send them to the service provider.

V. Conclusion

Current efforts in defining identity assurance in the open world show its importance for service providers as well as service consumers when interacting in a global market. Only if a relying party can assess the degree of confidence it can put in the assertion made by someone else, it will be able to accept this information and to use it in its own processes. Unfortunately, existing identity assurance frameworks assess identity providers mostly as a whole which leads to the situation, that identity attributes are mostly requested from a specific trusted identity provider if the trust requirements are high or from any identity provider, if there are no trust requirements. In our model, we aim at providing trust information on the level of identity attributes, especially about the verification process, and to use this information in policies and attribute requests to allow a more flexible choice of identity providers as well as an aggregation of identity attributes from multiple sources.

In this paper, we present our attribute assurance framework that augments identity management solutions with the ability to define trust in identity providers as well as trust into single attributes. We use an approach based on Horn logic to reason over a defined knowledge base containing trust information about identity providers, the attributes they can assert as well as the verification process. The approach is implemented in a Java-Prolog based framework and has been used in various example scenarios. Two use cases to show the applicability of the approach are described as part of this paper. In particular, at the identity provider side, we match requests for attributes with a particular verification need with the available attributes in the provider. At the service provider side, we use our framework during policy creation to find suitable identity providers that match the given trust requirements.

References

[1] Jøsang, A., & Pope, S. (2005). User Centric Identity Management. In A. Clark, K. Kerr, & G. Mohay (Ed.), AusCERT Asia Pacific Information Technology Security Conference, (p. 77).

[2] Rieger, S.: User-Centric Identity Management in Heterogeneous Federations. ICIW '09: Proceedings of the 2009 Fourth International Conference on Internet and Web Applications and Services (2009).

[3] Thomas, I. and Meinel,Ch.: "Enhancing Claim-Based Identity Management by Adding a Credibility Level to the Notion of Claims," Services Computing, IEEE International Conference on, pp. 243-250, 2009 IEEE International Conference on Services Computing, 2009

[4] Thomas,I. and Meinel, Ch.: An identity provider to manage reliable digital identities for SOA and the web. IDTRUST '10: Proceedings of the 9th Symposium on Identity and Trust on the Internet (2010).

[5] The OpenId Foundation: OpenID Authentication 2.0 - Final Specification, 2007,http://openid.net/specs/.

[6] OASIS: Identity Metasystem Interoperability Version 1.0, 2009, OASIS Standards.

[7] Anthony Nadalin et al.: WS-SecurityPolicy 1.2, OASIS Standard, July 2007.

[8] Office of the e-Envoy, UK: Registration and Authentication - e-Government Strategy Framework Policy and Guidelines, 2002.

[9] National Institute of Standards and Technology: Electronic Authentication Guideline, 2006.

[10] InCommon Federation., http://www.incommonfederation.org/

[11] InCommon Federation: Identity Assurance Profiles Bronze and Silver, 2010.

[12] Chadwick and Inman: Attribute aggregation in federated identity management. Computer (2009) vol. 42 (5) pp. 33-40.

[13] National Institute of Standards and Technology, http://www.nist.gov

[14] Mohan and Blough: AttributeTrust - A Framework for Evaluating Trust in Aggregated Attributes via a Reputation System. Sixth Annual Conference on Privacy, Security and Trust, pp. 201-212, 2008.

[15] Bhargav-Spantzel, A.: Protocols and Systems for Privacy Preserving Protection of Digital Identity, PhD Thesis (2007). http://www.gradschool.purdue.edu/downloads/ETDForm9-E2.pdf, 1-222.

[16] Paci, F., Bertino, E., Kerr, S., Squicciarini, A., Woo, J. (2009). An Overview of VeryIDX - A Privacy-Preserving Digital Identity Management System for Mobile Devices. Journal of Software, 4(7), 1-11. doi:10.4304/jsw.4.7.696-706

[17] Yong, J., Bertino, E.: *Digital identity enrolment and assurance support for VeryIDX.* Computer Supported Cooperative Work in Design (CSCWD), 2010 14th International Conference on, 734-739. doi:10.1109/CSCWD.2010.5471880

[18] Baldwin, A., Casassa Mont, M., Beres, Y., & Shiu, S.: Assurance for federated identity management. Journal of Computer Security - Digital Identity Management (DIM 2007) , 541-572, 2010.

[19] Identity, Credential & Access Management (ICAM): Trust Framework Provider Adoption Process (TFPAP) For Levels of Assurance 1, 2, and Non-PKI 3, Version 1.0.1, Release Candidate, Sept. 2009.

[20] Anthony Nadalin et al.: WS-Trust 1.3, OASIS Standard, March 2007.

[21] SAML V2.0, OASIS Standard, March 2005.

[22] Arun Nanda and Michael B. Jones: Identity Selector Interoperability Profile V1.5, July 2008.

[23] Michael B. Jones and Michael McIntosh: Identity Metasystem Interoperability Version 1.0, Committee Draft 01, November 2008.

[24] The OpenId Foundation: OpenID Attribute Exchange 1.0 - Final Specification, 2007,http://openid.net/specs/.

[25] Asir S. Vedamuthu et al.: Web Services Policy 1.5 - Framework, World Wide Web Consortium Recommendation, Sept. 2007.