SEVENTH FRAMEWORK PROGRAMME
Challenge 1
Information and Communication Technologies

Trusted Architecture for Securely Shared Services

Document type	Deliverable
Title	D4.1- TAS^3 Identifiers and Discovery
Work Package	WP2
Dissemination level	Public
Editor(s)	Sampo Kellomäki, Symlabs
Preparation date	30 June 2010
Version	05 (1.23)

Legal Notice

All information included in this document is subject to change without notice. The Members of the TAS3 Consortium make no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Members of the TAS3 Consortium shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

The TAS3 Consortium

	Participant name	Country	Participant short name	Participant role
1	K.U. Leuven	BE	KUL	Coordinator
2	Synergetics	BE	SYN	Partner
3	University of Kent	UK	KENT	Partner
4	University of Karlsruhe	DE	KARL	Partner
5	Technische Universiteit Eindhoven	NL	TUE	Partner
6	CNR/ISTI	IT	CNR	Partner
7	University of Koblenz-Landau	DE	UNIKOLD	Partner
8	Vrije Universiteit Brussel	BE	VUB	Partner
9	University of Zaragoza	ES	UNIZAR	Partner
10	University of Nottingham	UK	NOT	Partner
11	SAP Research	DE	SAP	Project Manager
12	Eifel	FR	EIF	Partner
13	Intalio	FR	INT	Partner
14	Risaris	IR	RIS	Partner
15	Kenteq	BE	KETQ	Partner
16	Oracle	UK	ORACLE	Partner
17	Custodix	BE	CUS	Partner

Contributors

	Name	Organisation
1	Brendan Van Alsenoy	KUL
2	Michele Bezzi	SAP
3	David Chadwick	KENT
4	Jeroen Hoppenbrouwers	SYN
5	Sampo Kellomäki (main contributor)	SYM
6	Gilles Montagnon	SAP

0 Table of Contents

Keyword List

Architecture, Identifiers, Security, Trust, Privacy

---

2 Identifiers and Discovery Executive Summary

This document describes identifier and token issuance considerations and services. It describes two principal categories of privacy friendly identifiers, the persistent and transient Name IDs that are difficult to guess and not shared across participants of a federation.

The data model of the federation databases is discussed and it is noted that the databases of an Identity Provider, discovery, linking service, and ID Mapper are highly similar and that a common implementation choice is to have the same system entity offer all these interfaces from a single database. However, to support separation of duties, an alternate model with separate databases and controlled synchronization is presented as well.

The issuance of tokens by an ID Mapper in various specific situations is discussed. The properties of the tokens and the necessary policy and audit safeguards are presented. We cover user-present, pre-authorized, and not-present cases as well as token based delegation.

A conclusion about token revocations is that most short term tokens do not need a revocation mechanism. In case of the Identity Mapper (IM) bootstrap token, which due to the logistics has to be long lived, specific risk mitigation strategies are adopted. In any case all derived tokens will be short lived and authorized upon token creation, effectively providing revocation of the IM bootstrap.

The role of the Registry Server in locating per-user resources is discussed. We also discuss how the Registry Server integrates with the On-line Compliance Testing and Trust Network's partner intake process.

Finally an exposition of the Credentials and Privacy Negotiation functionality is presented, including user interface driven front channel and discovery driven back channel approaches. Gap analysis is provided to see how the two phases of the back channel approach, discovery and service call, satisfy the essential needs to communicate policy pledges and policy requirements.

1 Introduction

This document specifies the TAS³ Discovery function, see [TAS3ARCH] Fig-2.2, comprising of ID Mapper, Registry Server, Linking, and Trust and Privacy Negotiator. The discovery function aims at solving two problems: issuance of credentials, or tokens, for specific transactions such that wild card credentials can be avoided; and finding out where a given service is hosted for given user, so that it is possible to host the same service for different users in different places, promoting competitive market place for the Service Providers.

This solution addresses Reqs. *D1.2-2.3-BMs* (discoverability), *D1.2-2.14-Priv* (pseudonymous design, attribute pull enablement), *D1.2-3.11-UPAPD* (the policy discovery aspect), *D1.2-7.17-Increm* (incremental release of credentials), *D1.2-3.12-SPManifest* (discovery based on privacy policy), *D1.2-3.13-BPAdapt* (business process adaptation by coordinating discovery), *D1.2-3.14-PIIPolicyDisco* (discovery keyed on adequate policies), *D1.2-3.15-SecPreserve* (discovery of policies so that business process can be adapted preserving certain policy properties), *D1.2-4.2-BPPrivacy* (use of pseudonyms in Business process).

An important architectural property pf the discovery function is that it allows fully pseudonymous operation, thus avoiding leakage of correlation handles and improving privacy protection in complex, intercalling, systems.

The discovery function also addresses user not present transactions, provides for some delegation scenarios, and acts as registry of services playing a part in Service Provider compliance validation business process.

1.1   Format and Properties of IDs

As specified in [TAS3PROTO] (also Annex A of [TAS3ARCH]), the primary token format of TAS³ is SAML 2.0 Assertion (A7N). In SAML 2.0, the users are identified by a *Name ID*, which can come in several variants. TAS³ chooses to use two principal kinds, see [SAML2core] sections 8.3.7 and 8.3.8, and attributes them (at least) the following properties:

Persistent Name ID Whenever the Identity Provider (IdP), or discovery, and federation partner talk about a user (e.g. IdP vouching authentication of the user to SP in a SSO transaction), they always use the same identifier, across the sessions. I.e. it can be understood that it is always the same user. Typically SP might maintain a database and use this identifier as a key.

Additional properties are required:

a. ID MUST be difficult to guess, ideally it should be at least 128 bit random number.

b. The ID used for same user towards different parties MUST NOT be easily inferable (e.g. it MUST NOT be same, statistically related, or guessable from the other ID).

Globally Unique ID or National ID would satisfy the permanence criteria, but this has adverse privacy consequences due to database linking.

The essence of our approach is that while a User may choose to (or be required to) allow one party to track his actions across sessions (hence persistent), this should not imply that this party can compare notes with other parties. The persistence

64 property allows tracking by legitimate party, while the "not easily inferable" property
65 keeps the parties from *colluding* or comparing notes.

66 The e-Government *sector specific* ID approach, e.g. tax number is different from social secu-
67 rity number and so forth for all government agencies, comes quite close to what we mean by
68 *persistent*.

69 Persistent Name ID is often called a *pseudonym*.

70 **Transient Name ID** This identifier format allows identification of the user for duration of one session.
71 It has the same properties as pseudonym, including (a) and (b) above, but will not persist across
72 sessions, thus providing better privacy guarantee than the Persistent Name ID, at the cost of
73 not allowing the SP to maintain a database. Often the motivation for using Transient Name ID is
74 exactly to prevent such databases from being created.

75 The most important privacy property, which both persistent and transient ID satisfy,
76 is that the User's identifier towards two different parties must be different and not
77 easily inferable. This provides a technical protection against cross site collusion. The
78 difference between the persistent and transient is that former allows the (authorized)
79 site to correlate across sessions, i.e. same user visiting same site repeatedly, while
80 the transient ID makes even this form of correlation difficult.

81 It is important to understand that the nonce and transient properties alone do not
82 provide significant privacy benefit if the random transient identifier is shared across
83 web sites even momentarily. Even single instance of such sharing would provide
84 the sites with a *correlation handle* despite the nonce and transient properties. Just
85 a single occurrence of a correlation handle allows all (persistent) past and future
86 click-trails to be linked across the web sites.

87 Other kinds of Name IDs are possible and allowed, depending on agreement within the Trust Net-
88 work. However, it should be fully understood what the privacy and other properties of the chosen ID
89 scheme are. TAS[3] has analyzed privacy and integrity of Persistent and Transient Name ID solutions.

1.2 Who Issues IDs

91 In TAS[3] the Name IDs that pass over the protocol flows are issued either by an IdP or the ID Mapper
92 (IM) of the discovery function. The internal IDs that SPs may have issued are generally not passed
93 over the wire (but see SAML 2.0 ManageNameID protocol [SAML2core] for a possible exception).

94 If the authentication scheme at the IdP involves a User ID, such a User ID is considered to be part of
95 the authentication credential. Allocation of the User IDs is private matter of the IdP and the User IDs
96 are never communicated to other parties or passed over the wire. It is possible that IdP collaborates
97 with some national eID scheme out side the scope of the TAS[3] architecture. In that case the eID would
98 probably be allocated by the national scheme, but the eID would not be communicated by IdP to the
99 Service Providers. The eID would only be used towards the IdP to authenticate the user and from that
100 point onwards the IdP allocated pseudonyms are used.

1.3 Supported Flows

Reader should refer to [TAS3ARCH], section 3 "Core Security Architecture" for description of the supported protocol flows.

104

2 Federation Data Model

One of the fundamental principles of the identifier management is use of federations. In order to implement persistent and pseudonymous federations, the IdP and IM have to keep state.

In general federation table for IdP has mappings of form

```
User at IdP1 --> [ encrypted pseudonym of user at SPA,
                   encrypted pseudonym of user at SPB,
                   ...
                   encrypted pseudonym of user at SPN ]
```

If the table serves only one IdP and thus the IdP EntityID is implicitly known, then the table simplifies to have columns

```
User ID AuthN Cred SP EntityID    Enc. pseudonym of user at SP
------- ---------- --------------  ----------------------------
Koerkki salainen   A.example.com   enc_A(123)
--''--  --''--     B.example.com   enc_B(456)
--''--  --''--     C.example.com   enc_C(246)
--''--  --''--     IM.example.com  enc_IM(789)
Tester  secret     A.example.com   enc_A(357)
--''--  --''--     IM.example.com  enc_IM(579)
```

where "enc_A", etc., means encryption such that only A can decrypt (e.g. with A's public key or shared secret only known to A). The encryption should also include a nonce component to avoid two encryptions of the same data looking the same. This is to protect the pseudonyms against exposure at middlemen and while in the database.

The federation table for IM needs similar mappings

```
User's pseudonym at IM --> [ encrypted pseudonym of user at SPA,
                             encrypted pseudonym of user at SPB,
                             ...
                             encrypted pseudonym of user at SPN ]
```

If the table serves only one IM, then the table simplifies to have columns

```
User's pseudonym SP EntityID    Enc. pseudonym of user at SP
---------------- --------------  ----------------------------
789IM            B.example.com   enc_B(456)
789IM            C.example.com   enc_C(246)
579IM            B.example.com   enc_B(791)
579IM            C.example.com   enc_C(913)
```

The same table format works for IM, and Linking Service.

The IdP and IM may include attribute data in the tokens they emit. This attribute data can be kept in any suitable data structure, usually indexed by user and sometimes by SP, or both.

The IM needs additional data structure to determine what services are available to a User. In its simplest form this would consist of

```
144   User's pseudonym Service Type SP EntityID
145   ---------------- ------------ -------------
146   789IM            Role Authr   C.example.com
147   789IM            HR Authr     B.example.com
148   579IM            Role Authr   C.example.com
149   579IM            HR Authr     B.example.com
```

150 but other more general realisations can include data needed for Credentials and Privacy Negotiation
151 phase of Discovery. These will be explored in Credentials and Privacy Negotiatior documentation.

152 An IdP may have a limited form of this table to cover the necessity of emitting IM bootstrap token
153 during SSO.

154 All parties - IdP, IM, and SP (Front End or Web Service) - need to maintain some metadata about
155 each other. Such metadata may include SOAP endpoints, protocol profiles and bindings to use, etc.,
156 see [SAML2meta].

3 ID Mapper: Issuing Specific Tokens

As specified in [TAS3PROTO], The ID Mapper functionality is realised as part of Discovery Service [Disco2]. It MAY also be realised using Security Token Service (STS) role of [WSTrust] or Identity Mapping Service described in [SOAPAuthn2] (this being typical in some delegation cases using People Service [PeopleSvc]).

The tokens issued by ID Mapper often pass through intermediaries due to the logistics of the discovery and token based delegation flows. To maintain fully pseudonymous architecture, the tokens that as passed through intermediaries MUST be encrypted using public key of the intended consumer of the token.

3.1 User Present Case

User present scenario is the base case of the TAS^3 architecture. It assumes the user is interacting with the system in (near) real time and instructs it to act according to his wishes. Such instruction simultaneously provides command of action and consent to it being performed, including any sub-actions that may be needed for performance. From the audit trail perspective, it is essential that User's manifest will and consent is captured. The audit trail becomes stronger when it can be shown that the user was tactically present and aware of the action taken. Presence is of course relative when a web service call is made somewhere deep in the infrastructure, but if it can be shown that user had an active front channel session and that from this session emanated a command to perform an action which caused the audited action to be performed, the presenceness of the user in the audited action is established.

In the user present case the token issuance is straight forward: the IM bootstrap token is generated by the IdP and included in the Single Sign-On (SSO) or web services layer authentication as an attribute in the assertion, as seen in Fig-3.1
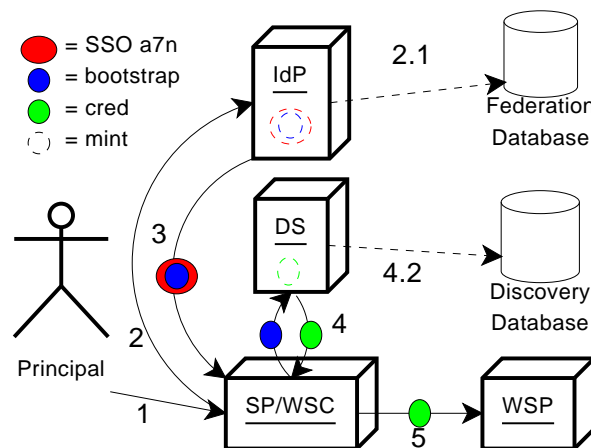

Figure 3.1: Single Sign-On (2,3), Discovery (4), and call to WSP (5). The blue ball represents discovery bootstrap.

The Service Provider consumes the SSO or authentication assertion and extracts the IM Bootstrap. When it needs to call a web service, it uses the IM Bootstrap to call the Id Mapper service, which

usually is a Liberty Discovery Service, but could also be a WS-Trust Security Token Service (STS), to generate the access token which is then used towards the payload web service.

3.1.1 User Present Evidence

The "user present" aspect is captured by the fact that the IdP attests direct authentication of the user in the same session and in a not too distant past. Generally the expiry time of the tokens is set accordingly.

This works well under the assumption that the web services call happens relatively soon after the SSO, but fails to provide adequate solution for long-lived SP sessions. For long lived sessions, the temptation is to increase the expiry time of the IM bootstrap token, but this weakens the "user present" aspect and ultimately some cut-off must be determined in the Governing Agreement of the Trust Network.

A better alternative, is to refresh the IM bootstrap by performing a new SSO transaction with the IdP: as long as user's session at the IdP is still valid, the refresh will not cause any user observable effect (apart from the redirection flicker), but will return a new IM bootstrap with renewed expiry time.

3.1.2 Interaction of IdP and ID Mapper

Since IdP and ID Mapper (IM) need to maintain federation databases and need to communicate with one-another for purposes of arranging the IM bootstrap, it is not uncommon for same organization to operate both IdP and IM so they can share a database. The Liberty Identity Mapping Service is also often co-hosted with the IdP and discovery and shares their database. In the shared database case, any suitable arrangement can be used and the standards tend to be silent on the issue.

If, however, there is desire to keep the databases of IdP and IM separate, e.g. for separation of duties purposes, then some communication needs to happen between IdP and IM to arrange the bootstrap issuance - which necessitates that the IdP learns the User's pseudonym at the IM.
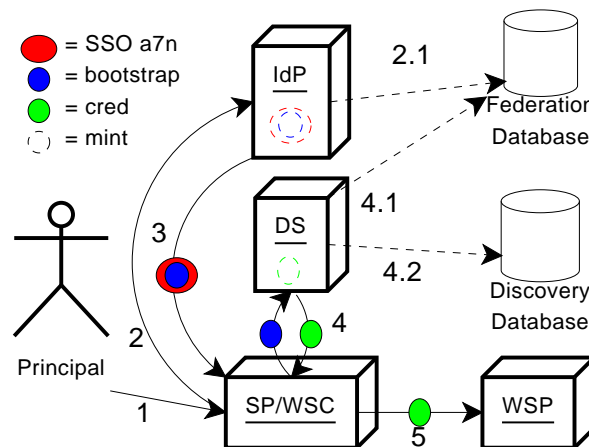

Figure 3.2: Discovery Service makes back channel query (4.1) to map the Id received in the bootstrap to a key that can be used to query other databases (4.2)

Fig-3.2 depicts a scheme where the IM updates the IdP database. This is advantageous when IdP is a COTS software package that can not be altered. It does require a documented database interface,

₂₀₈ though. Many IdP products use LDAP repository as a database and work in straight forward way.
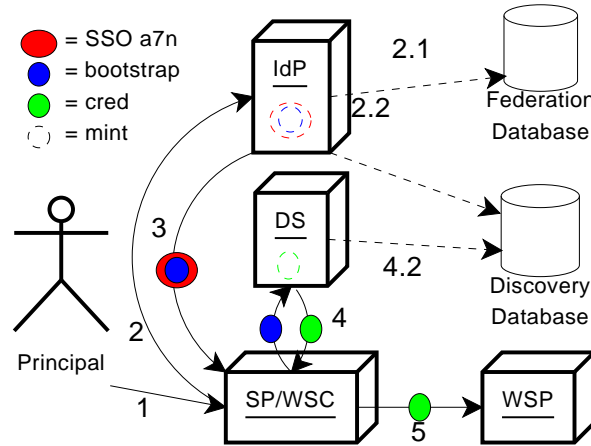₂₀₉ Currently there are no standards regarding the database schema.

Figure 3.3: IdP makes back channel query (2.2) to map the msisdn to Id understood by the discovery (4.2). The Id is passed in the bootstrap.

₂₁₀   Fig-3.3 shows the opposite arrangement where the IdP consults the IM database to obtain the IM
₂₁₁ bootstrap.

3.2 Pre-authorization by the User

₂₁₃ The pre-authorization case involves the user being present at some point in time and indicating his will
₂₁₄ that in future some operation is performed on his behalf. Such consent and authorization is captured
₂₁₅ by technical means so that the transaction can be acted in a later time producing an audit trail that
₂₁₆ unequivocably demonstrates that the user authorized the transaction, albeit in (possibly distant) past
₂₁₇ and without being present at the moment of the transaction.

₂₁₈   The pre-authorized case can easily be implemented by issuing a long-lived token that can then be
₂₁₉ used as if the user was still present. This is, however, suboptimal since the user looses control. Since
₂₂₀ implementing a revocation check and list for tokens is burdensome, we adopt an approach where the
₂₂₁ only long lived token that can be cached is the IM bootstrap. The pre-authorized case is implemented
₂₂₂ by using the IM bootstrap to obtain a token just prior to the pre-authorized transaction. The request for
₂₂₃ token shall claim the pre-authorized situation and the discovery will check appropriate policies to see
₂₂₄ if the contemplated pre-authorized transaction can go forward. This represents a depth of defence as
₂₂₅ a similar check can and should be made at the Service Provider.

₂₂₆   The identifiers used in the pre-authorized case can be of the same fully pseudonymous variety as in
₂₂₇ the user-present case. If pre-authorization is attributed with an identifier (e.g. authorization number),
₂₂₈ care should be taken to avoid this identifier from turning into a correlation handle.

3.3 User Not Present Case

₂₃₀ The user not present cases generally involve situations where legal or contractual justification can be
₂₃₁ invoked to authorize a transaction to go forward. The TAS[3] authorization infrastructure MUST check

²³² policies to determine if indeed it is legally or contractually acceptable to perform any give transaction
²³³ without user's consent or presence.

²³⁴ An interesting identity management problem that arises in the user-not-present case is how the user
²³⁵ can be accurately identified. In case of globally unique ID, this may be relatively easy, but in case of
²³⁶ fully pseudonymous identity management, the authorized initiator of the transaction may not actually
²³⁷ know the pseudonyms that are needed to complete the transaction.

²³⁸ To resolve this impasse, the identity mapping functionality described in Section 3.5 can be used
²³⁹ to convert a pseudonym that the initiator knows to a pseudonym that he needs. As such generic
²⁴⁰ conversion ability is a serious privacy threat, the authentication of the initiator and authorization must
²⁴¹ be of the highest standards. The Trust Network should invest significant planning and audit to make
²⁴² sure that the identity mapping does not become dangerous backdoor.

²⁴³ If the initiator does not know any pseudonym, there are grounds to suspect that it does not have a
²⁴⁴ legitimate case for performing a user-not-present -transaction. If such transaction is legitimate, none-
²⁴⁵ the-less, then initiator judicially requests a pseudonym for the user to be disclosed by a party (identity
²⁴⁶ discloser) that can check the legitimacy of the request and understand which user is meant. Such
²⁴⁷ identity discloser could also provide search and browsing interfaces for finding the user. The caveat
²⁴⁸ about grave privacy and security concerns mentioned in identity mapping, above, doubly apply to the
²⁴⁹ identity discloser.

²⁵⁰ All tokens emitted via identity discloser MUST be marked as such and should not purport to be user
²⁵¹ present tokens. The authorization at the SP in this case is based on the legitimate-user-not-present-
²⁵² access marker. The initiating party MUST be identified in each user-not-present transaction. This
²⁵³ could be achieved by regular authentication of the initiator using authentication mechanisms specified
²⁵⁴ in [SOAPBinding2] or it could be expressed using Subject Identity in token based delegation.

²⁵⁵ Once the user-not-present transaction in principle is authorized, the problem remains as to which
²⁵⁶ type of token should be issued to the initiator. To keep authorization narrow, the token should be
²⁵⁷ directly destined to the SP to which access is authorized. However, as there is no easy way to know
²⁵⁸ if the SP in its turn needs to call on other SPs to perform its function, it is necessary to provide an IM
²⁵⁹ bootstrap as well.

²⁶⁰ Problem with providing an IM bootstrap in user-not-present transaction is that it may authorize too
²⁶¹ wide access. This is an active research topic in TAS³ and we hope to present a solution in a future
²⁶² revision of this deliverable. Meanwhile, the best that can be done is to ensure extended audit to detect
²⁶³ and curtail any abuse.

²⁶⁴ 3.4 User Revokes Authorization

²⁶⁵ Revocation of authorization should mainly happen through policy mechanisms.

²⁶⁶ It is, however, possible to revoke the identity associated with the authorization. Revocation (or
²⁶⁷ suspension) of identity prevents further issuance of tokens. Thus the problem that remains is what to
²⁶⁸ do with the already existing tokens. Currently (May 2009) TAS³ does not foresee a token revocation
²⁶⁹ mechanism. The intent is that all issued tokens are so short lived that revocation check is not needed.

²⁷⁰ The notable exception is the pre-authorized case. As described in Section 3.2, this is solved by only
²⁷¹ tolerating long lived IM bootstrap token. All other tokens are issued from this bootstrap on as-needed
²⁷² basis and are quickly expiring. This achieves the same net effect as revocation list check.

²⁷³ It should be noted that for PKI certificates we do foresee use of OCSP [RFC2560] or revocation lists.
²⁷⁴ However as PKI is not used for end user identity (except, perhaps by IdP to authenticate the user), this

275 revocation processing is not in scope of this document.

3.5 Supporting Delegation

277 The delegation flow depicted in Fig-3.14 of [TAS3ARCH] includes steps 5a-5b where the Delega-
278 tee's identity at the Delegation Service is converted to his identity at the SP. This conversion step is
279 necessary to allow fully pseudonymous identity, i.e. the Delegation Service and SP will not share a
280 correlation handle.

281 The identity mapping function is realised using the interface described in Section 7 "Identity Mapping
282 Service" of [SOAPAuthn2]. For it to function, it needs to maintain a federation database that is very
283 similar to the one already maintained for discovery and IdP purposes. A common implementation and
284 deployment choice is to co-locate the identity mapping functionality with the discovery and IdP, sharing
285 the databases. If it is desirable, e.g. for separation of duty reasons, to keep the identity mapping
286 separate, then some database synchronization scheme is needed, perhaps similar to ones described
287 in Section 3.1.2, above.

4 Linking Service

The identity management ramifications of the Linking Service are still subject to research. It would appear that privacy will be difficult to preserve in linking service as a single, albeit random, identifier is shared between several entities. We will elaborate on this in a future version of this document.

5 Registry Server

The registry server is part of the discovery functionality. Its purpose is to maintain a database of Service Providers (SPs), a mapping of which SP provides which service to which user, and a federation database specifying the User's (encrypted) pseudonym at each SP with which the user has a relationship.

Such database is needed to ensure that Users have a choice of SP or at least to ensure that different Users can get the service from disjoint sets of SPs. Such situations commonly arise when Users of one organization start using services of another organization. For example, if User uses procurement application at Supplier, the User's roles still need to be fetched from his home organization. The registry server enables the supplier to dynamically understand where the role authority is for each of the Users that use the system.

5.1 On-line Registration Steps

Given the fully pseudonymous design of the TAS^3 architecture, populating the registration database is non-trivial. Fig-5.1 presents a dynamic solution where the User initiates the registration. In this case the (encrypted) pseudonym for the user at the SP is pushed to the registry by the SP itself.
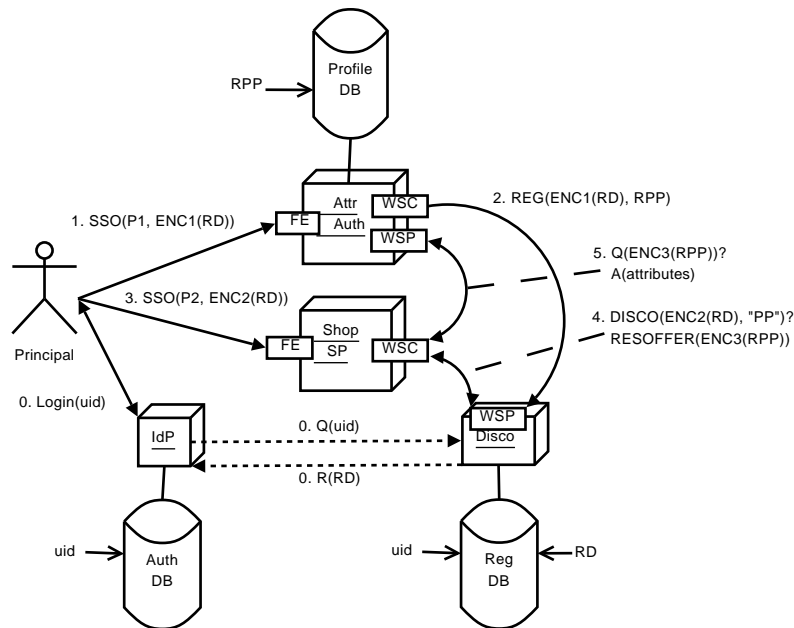

Figure 5.1: Discovery Registration Using Front Channel Interface.

The registration steps are as follows:

1. User visits a service, perfoming a Single Sign-On, thus establishing his pseudonymous identity at the service.

2. User triggers the service to register itself as one of the user's services. At this point the discovery database records what it should send as users identity in a subsequent web service call.

314 3. User instructs the front end to perform an action that triggers a web service call.

315 4. First the discovery step is made to obtain the token.

316 5. The actual web service call is made with the correct identity.

5.2 Bulk Registration

318 The dynamic registration model may not be appropriate in all situations. Bulk registration offers an
319 alternative, but presents a problem as populating the registration database will require identification of
320 the Users, i.e. the pseudonym of the user at each SP needs to be found. The problem is similar to the
321 user-not-present case, see Section 3.3. The technical solution is essentially same as in Section 3.1.2.

5.3 Articulation of Registration with Compliance Validation

323 The Registry Server plays an important role in the On-line Compliance Testing as it is the mecha-
324 nism by which the testing infrastructure finds out about new system entities to test and their relevant
325 metadata, test cases, and declared policies.
326 The Trust Network level new organization intake process integrates a registration step.

6 Credentials and Privacy Negotiation

The Credentials and Privacy Negotiation (CPN) can happen at front channel or at back channel. In the former case it can involve choice of Front End and mutually assuring steps shown through the user interface. The purpose of the steps is to climb a ladder of trust where each party progressively reveals more about itself until trust can be established between the parties. Front channel Credentials and Privacy Negotiation is user interface intensive and may need to be modelled at business process level. As it is likely to be extremely implementation and deployment specific, it is not discussed any further here.
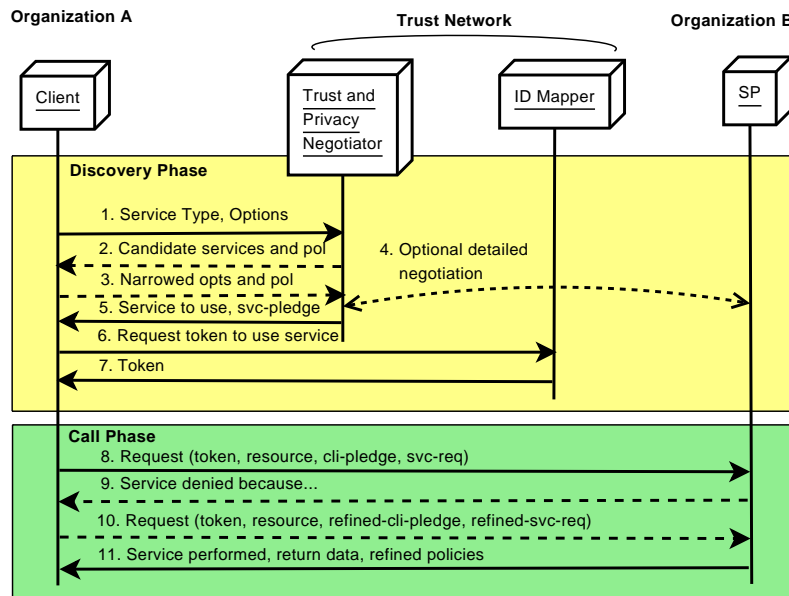

Figure 6.1: Two phase Credentials and Privacy Negotiation on back channel.

The CPN on back channel involves stating trust and privacy requirements at both the client and service side and then discovering a service that matches the client. If a single match is found, the negotiation ends and the service provider is used. If multiple matches are found, the client may unilaterally choose the one that is most advantageous to it.

If no matches are found, the real negotiation begins. The client may relax some of its requirements and see if it can discover any SPs under the new terms. Alternatively, the discovery may return some SPs whose policies are close enough that it seems the client might relax its requirements sufficiently, with the trust and privacy parameters they accept, and the client then has to make a new query, promising to satisfy the refined parameters that were required, to obtain the access credentials.

Final part of the Credentials and Privacy Negotiation can be carried once the client has chosen a service. The service request contains a request specific pledge by the client about the policies it promises to honour, if the request is performed, and with respect to the data that may be returned. The SP examines this policy pledge and decides if it is acceptable given the request and the data it would return.

If policies are acceptable, the SP performs the request, attaching to the response additional policies that the client must honour. These policies can be only ones that are foreseen by (a) law, (b) contract, or (c) client's policy pledge carried in the request. For example, if client's policy pledge promises to

honour any data retention limitation above 5 seconds, then the service could set an obligation to delete the returned data in 60 seconds. Insisting on deletion in 3 seconds would be moot as the client never promised to honour that.

If the policy pledge of the Client is not acceptable, the service returns an error indicating why it was not acceptable or what would have been acceptable. The Client can then decide if it is willing to modify its policy pledge until it is acceptable and retry, or abandon the request. The Client can then return to the discovery phase and try to locate a different Service Provider candidate.

There are two key decisions in the last phase: (i) the Service Provider needs to decide whether the Client's policy pledge is acceptable given the nature of the request, the User behind the request, including any delegation, and the trustworthiness of the Client itself; and (ii) if Service Provider tries to negotiate on the policy pledge, the Client needs to decide whether it finds the proposed pledge acceptable given the trustworthiness of the Service Provider.

In both of these cases the trustworthiness of the other party needs to be established. This is primarily done using TAS^3 trust establishment mechanisms, which tend to communicate whether the Trust Network considers the other party a trustworthy participant of the network. If the parties have specific trust requirements, beyond what the Trust Network is able to tell about the parties, then they need to do some extra work themselves. Often such specific trust evaluation will be specific to the business of the Client and/or the Service Provider, thus it is expected that they will establish their own business processes for it. For example, Service Provider may have a requirement that the mere Trust Network membership is not sufficient, thus it needs to invoke the Service Provider specific Client Intake business process to get the Client vetted and registered. If this happens in the background on the Service Provider side, the main flow is not affected. If, however, the Client Intake business process needs to be initiated by the Client, then the Service Provider needs to return an error code or policy requirement that triggers the Client to initiate the process. In this case there needs to be a private understanding about the meaning of these error codes between the Client and the Service Provider. TAS^3 foresees this mechanism, but does not specify what the codes are.

6.1 Expressing Policy Pledges and Requirements

In a contemplated transaction between Client and Service Provider (SP) following policy requirements and pledges need to be passed

A. What policies Client pledges to honour

B. What policies Client requires SP to honour

C. What policies SP pledges to honour

D. What policies SP requires Client to honour

The Credentials and Policy Negotiation is always initiated by the Client, but tends to be driven by responses of the Service Provider. The protocols at discovery and service call levels need to support passing policy pledges from Client to Service Provider and policy requirements from Service Provider to the Client.

6.1.1 Expressing Pledges and Requirements in Discovery Phase

Some trust and privacy requirements can be expressed as [CARML] declarations, permitting fairly fine grained specification of the needs of the Client and the policies it is willing to respect. [AAPML] declaration allows a service to express its policy requirements so that Clients know what they need to honour if they plan to communicate with the service. Both CARML and AAPML work best during the discovery phase (and discovery registration).

Discovery options, see [Disco2], are another mechanism for expressing what policy requirements the candidate Service Providers for the contemplated transaction must satisfy.

Given the 4 requirements stated in beginning of this Section 6.1, the gap analysis against [Disco2] indicates deficiency in (A) and good support for (B) and (C). Supporting (D) can be easily achieved by adding a processing rule that the discovery should return candidate results that do not entirely match so that the Client can get an idea of what would be acceptable.

In our current (May 2009) thinking, the gap (A) is left open at discovery phase and will be addressed in the call phase. If the discovery really needs to know the Client's pledge, then [CARML] could be used. [AAPML] is superb for expressing (D).

6.1.2 Expressing Pledges and Requirements in Call Phase

In the call phase the Client effectively keeps on trying the transaction with different policy pledges until it succeeds. In the call, client uses `<UsageDirective>` element, see [SOAPBinding2], to convey its policy pledge.

The Client expresses its acceptance of the negotiation by performing a service request. If it has not obtained an acceptable policy from the Sp, then it can not make the request. Currently (May 2009) there is no mechanism for Client to request relaxation of the policy, other than just try request with policy that is formally known to be unacceptable to the SP. More elegant solution to this situation is an area of active TAS[3] research.

The SP expresses its acceptance by performing the service and returning refined policies as `<UsageDirective>`. If it does not accept, the error returns (combined with `<UsageDirective>`) provide a readily available way of conveying policy requirements.

Given the 4 requirements stated in beginning of this Section 6.1, the gap analysis against [SOAPBinding2] indicates that (A) is well covered by `<UsageDirective>`. Specification of (B) is not expressly called out, but the `<UsageDirective>` in the request can be used for this purpose as well. In call phase there is no support for (C), but discovery phase supports it relatively well. Only problem is that the discovery phase service pledge can not take the specific resource and operation in account. If this really turns out to be a requirement, future versions of TAS[3] architecture may specify some mechanism for doing this. Support for (D) is two fold: the SP sees from request `<UsageDirective>` what the Client pledging and can then include in response a `<UsageDirective>` to refine the requirements. If the SP wishes to impose requirements that are not mere refinements, then it must refuse the service call and provide a `<UsageDirective>` expressing what it would have accepted.

6 Bibliography

[AAPML]   Prateek Mishra, ed.: "AAPML: Attribute Authority Policy Markup Language", Working Draft 08, Nov. 28, 2006, Liberty Alliance / Oracle. http://www.oracle.com/technology/tech/standards/idm/igf/pdf/IGF-AAPML-spec-08.pdf

[CARML]   Phil Hunt and Prateek Mishra, eds.: "Liberty IGF Client Attribute Requirements Markup Language (CARML) Specification", Draft 1.0-12, Liberty Alliance, 2008. http://www.projectliberty.org/liberty/resource_center/specifications/igf_1_0_specs

[Disco2]   Cahill, ed.: "Liberty ID-WSF Discovery service 2.0", liberty-idwsf-disco-svc-2.0-errata-v1.0.pdf from http://projectliberty.org/resource_center/

[PeopleSvc] "Liberty ID-WSF People Service Specification", liberty-idwsf-people-service-1.0-errata-v1.0.pdf from http://projectliberty.org/resource_center/specifications/

[RFC2560]   Myers et al., "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[SAML2core] "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", Oasis Standard, 15.3.2005, saml-core-2.0-os

[SAML2meta] Cantor, Moreh, Phipott, Maler, eds., "Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0", Oasis Standard, 15.3.2005, saml-metadata-2.0-os

[SOAPAuthn2] "Liberty ID-WSF Authentication, Single Sign-On, and Identity Mapping Services Specification", liberty-idwsf-authn-svc-2.0-errata-v1.0.pdf from http://projectliberty.org/resource_center/specifications/

[SOAPBinding2] "Liberty ID-WSF SOAP Binding Specification", liberty-idwsf-soap-binding-2.0-errata-v1.0.pdf from http://projectliberty.org/resource_center/specifications

[TAS3ARCH]   Sampo Kellomäki, ed.: "TAS3 Architecture", TAS3 Consortium, 2009. Document: tas3-arch-vXX.pdf

[TAS3PROTO] Sampo Kellomäki, ed.: "TAS3 Protocols and Concrete Architecture", TAS3 Consortium, 2009. Document: tas3-proto-vXX.pdf

[WSTrust]   "WS-Trust 1.3", CD 6, OASIS, Sept 2006. (*** WS-Trust, STS, etc.)

Document ID `tas3-disco-v05.pdf`

URL path `https://portal.tas3.eu/arch/review/tas3-deliv-4_1-disco-v05.pdf`