

SEVENTH FRAMEWORK PROGRAMME
Challenge 1
Information and Communication Technologies



Document type	Deliverable
Title	D4.1- TAS ³ Identifiers and Discovery
Work Package	WP2
Dissemination level	Public
Editor(s)	Sampo Kellomäki, EIfEL
Preparation date	1 June 2009
Version	04 (1.21)

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)	EIF
6	Gilles Montagnon	SAP



0 Table of Contents

EXECUTIVE SUMMARY	4
1 INTRODUCTION	5
1.1 FORMAT AND PROPERTIES OF IDS	5
1.2 WHO ISSUES IDS	6
1.3 SUPPORTED FLOWS	7
2 FEDERATION DATA MODEL	8
3 ID MAPPER: ISSUING SPECIFIC TOKENS	10
3.1 USER PRESENT CASE	10
3.1.1 User Present Evidence	11
3.1.2 Interaction of IdP and ID Mapper	11
3.2 PRE-AUTHORIZATION BY THE USER.....	12
3.3 USER NOT PRESENT CASE.....	12
3.4 USER REVOKES AUTHORIZATION.....	13
3.5 SUPPORTING DELEGATION	14
4 LINKING SERVICE	15
5 REGISTRY SERVER	16
5.1 ON-LINE REGISTRATION STEPS	16
5.2 BULK REGISTRATION	17
5.3 ARTICULATION OF REGISTRATION WITH COMPLIANCE VALIDATION	17
6 TRUST AND PRIVACY NEGOTIATION	18
6.1 EXPRESSING TRUST AND POLICY PLEDGES AND REQUIREMENTS	19
6.1.1 Expressing Pledges and Requirements in Discovery Phase	19
6.1.2 Expressing Pledges and Requirements in Call Phase	20
BIBLIOGRAPHY	21

Keyword List

1 Architecture, Identifiers, Security, Trust, Privacy

2 Identifiers and Discovery Executive Summary

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

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

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

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

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

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



25

26 1 Introduction

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

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

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

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

46 1.1 Format and Properties of IDs

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

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

55 Additional properties are required:

- 56 a. ID MUST be difficult to guess, ideally it should be at least 128 bit random number.
- 57 b. The ID used for same user towards different parties MUST NOT be easily inferable (e.g. it
58 MUST NOT be same, statistically related, or guessable from the other ID).

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

61 The essence of our approach is that while a User may choose to (or be required to)
62 allow one party to track his actions across sessions (hence persistent), this should
63 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. TAS³ has analyzed privacy and integrity of Persistent and Transient Name ID solutions.

90 1.2 Who Issues IDs

91 In TAS³ 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 TAS³ 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.



101 **1.3 Supported Flows**

102 Reader should refer to [\[TAS3ARCH\]](#), section 3 "Core Security Architecture" for description of the sup-
103 ported protocol flows.

104

105 2 Federation Data Model

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

108 In general federation table for IdP has mappings of form

```
109 User at IdP1 --> [ encrypted pseudonym of user at SPA,  
110                   encrypted pseudonym of user at SPB,  
111                   ...  
112                   encrypted pseudonym of user at SPN ]
```

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

115	User ID	AuthN Cred	SP	EntityID	Enc. pseudonym of user at SP
116	-----	-----	-----	-----	-----
117	Koerkki	salainen		A.example.com	enc_A(123)
118	--''--	--''--		B.example.com	enc_B(456)
119	--''--	--''--		C.example.com	enc_C(246)
120	--''--	--''--		IM.example.com	enc_IM(789)
121	Tester	secret		A.example.com	enc_A(357)
122	--''--	--''--		IM.example.com	enc_IM(579)

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

127 The federation table for IM needs similar mappings

```
128 User's pseudonym at IM --> [ encrypted pseudonym of user at SPA,  
129                             encrypted pseudonym of user at SPB,  
130                             ...  
131                             encrypted pseudonym of user at SPN ]
```

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

133	User's pseudonym	SP	EntityID	Enc. pseudonym of user at SP
134	-----	-----	-----	-----
135	789IM		B.example.com	enc_B(456)
136	789IM		C.example.com	enc_C(246)
137	579IM		B.example.com	enc_B(791)
138	579IM		C.example.com	enc_C(913)

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

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

142 The IM needs additional data structure to determine what services are available to a User. In its
 143 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 Trust and Privacy Negotiation phase
151 of Discovery. These will be explored in Trust and Privacy Negotiator 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

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

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

3.1 User Present Case

168 User present scenario is the base case of the TAS³ architecture. It assumes the user is interacting
 169 with the system in (near) real time and instructs it to act according to his wishes. Such instruction
 170 simultaneously provides command of action and consent to it being performed, including any sub-
 171 actions that may be needed for performance. From the audit trail perspective, it is essential that User's
 172 manifest will and consent is captured. The audit trail becomes stronger when it can be shown that
 173 the user was tactically present and aware of the action taken. Presence is of course relative when a
 174 web service call is made somewhere deep in the infrastructure, but if it can be shown that user had
 175 an active front channel session and that from this session emanated a command to perform an action
 176 which caused the audited action to be performed, the presenceness of the user in the audited action
 177 is established.

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

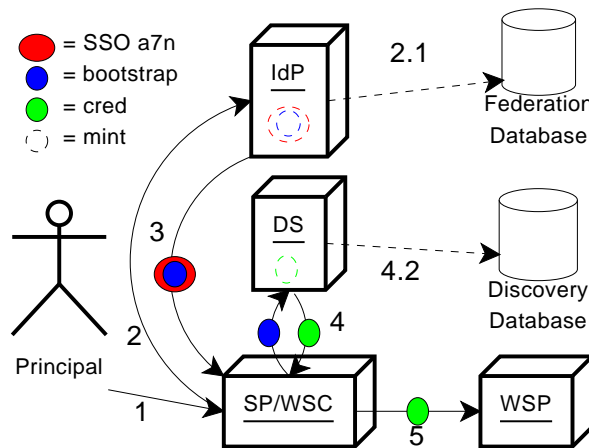


Figure 3.1: Single Sign-On (2,3), Discovery (4), and call to WSP (5). The blue ball represents discovery bootstrap.

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

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

185 **3.1.1 User Present Evidence**

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

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

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

197 **3.1.2 Interaction of IdP and ID Mapper**

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

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

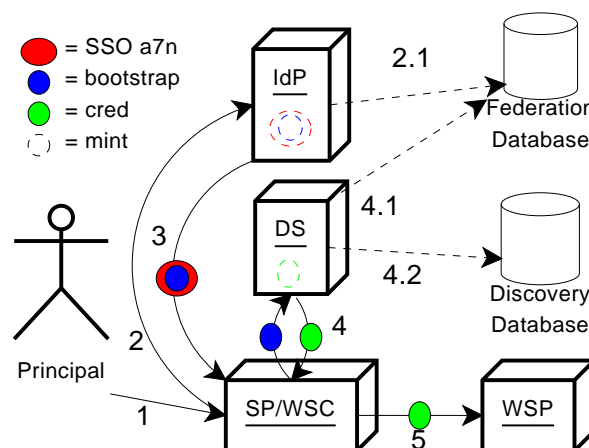


Figure 3.2: Discovery Service makes back channel query (4.1) to map the Id received in the bootstrap to a key that can be used to query other databases (4.2)

206 Fig-3.2 depicts a scheme where the IM updates the IdP database. This is advantageous when IdP
 207 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.

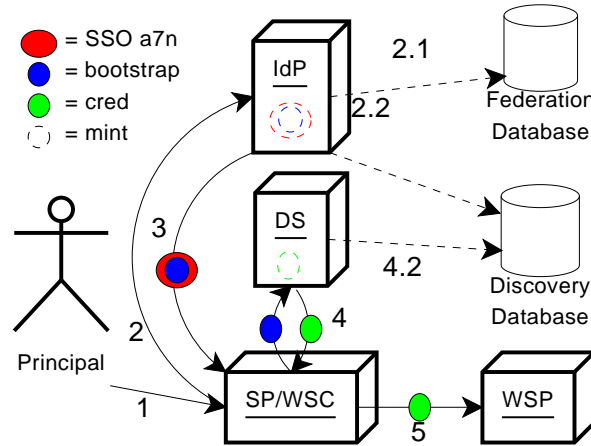


Figure 3.3: IdP makes back channel query (2.2) to map the msisdn to Id understood by the discovery (4.2). The Id is passed in the bootstrap.

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

212 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 unequivocally 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 loses 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.

229 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 TAS³ 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.

276 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.



288

289 4 Linking Service

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

5 Registry Server

294

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

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

5.1 On-line Registration Steps

305

306 Given the fully pseudonymous design of the TAS³ architecture, populating the registration database is
 307 non-trivial. Fig-5.1 presents a dynamic solution where the User initiates the registration. In this case
 308 the (encrypted) pseudonym for the user at the SP is pushed to the registry by the SP itself.

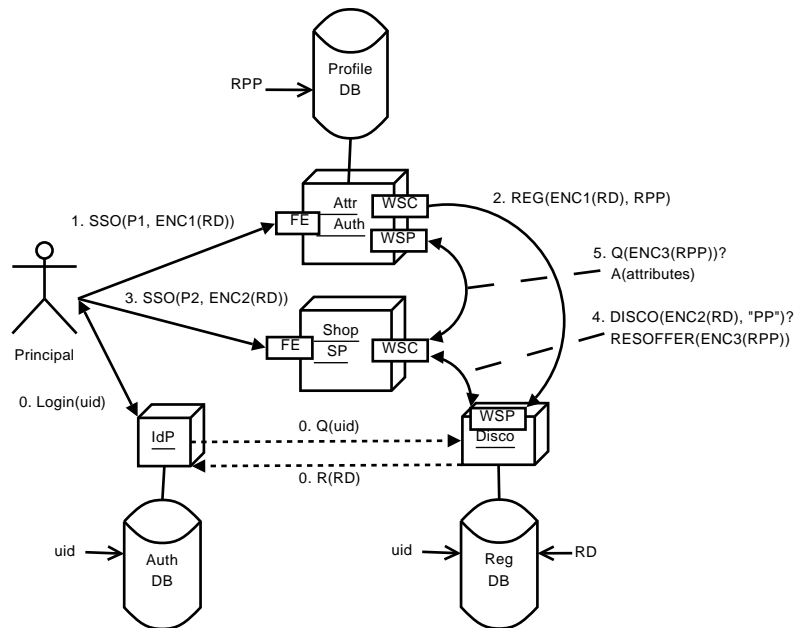


Figure 5.1: Discovery Registration Using Front Channel Interface.

309 The registration steps are as follows:

- 310 1. User visits a service, performing a Single Sign-On, thus establishing his pseudonymous identity at
 311 the service.
- 312 2. User triggers the service to register itself as one of the user's services. At this point the discovery
 313 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.

317 **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.

322 **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 Trust and Privacy Negotiation

328

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

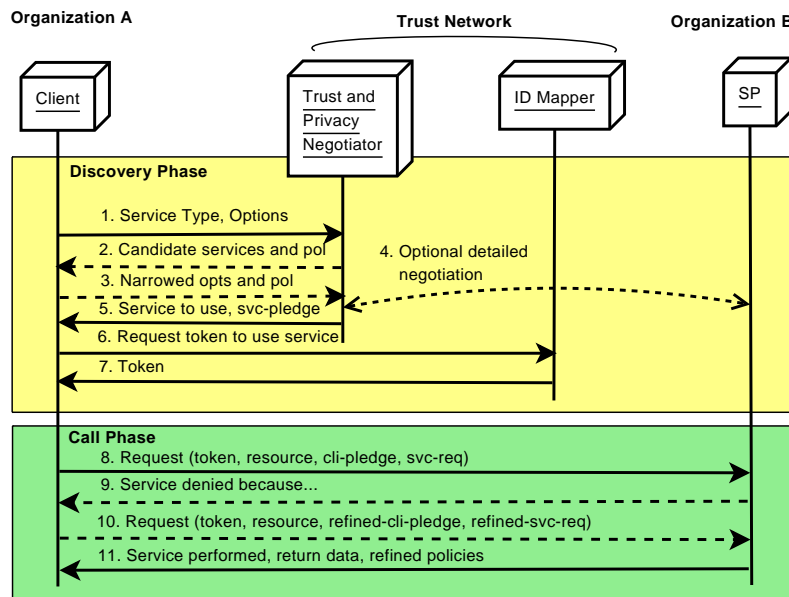


Figure 6.1: Two phase Trust and Privacy Negotiation on back channel.

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

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

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

349 If policies are acceptable, the SP performs the request, attaching to the response additional policies
 350 that the client must honour. These policies can be only ones that are foreseen by (a) law, (b) contract,
 351 or (c) client's policy pledge carried in the request. For example, if client's policy pledge promises to
 352 honour any data retention limitation above 5 seconds, then the service could set an obligation to delete



6.1. EXPRESSING TRUST AND POLICY PLEDGES AND REQUIREMENTS

353 the returned data in 60 seconds. Insisting on deletion in 3 seconds would be moot as the client never
354 promised to honour that.

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

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

364 In both of these cases the trustworthiness of the other party needs to be established. This is primar-
365 ily done using TAS³ trust establishment mechanisms, which tend to communicate whether the Trust
366 Network considers the other party a trustworthy participant of the network. If the parties have specific
367 trust requirements, beyond what the Trust Network is able to tell about the parties, then they need to
368 do some extra work themselves. Often such specific trust evaluation will be specific to the business
369 of the Client and/or the Service Provider, thus it is expected that they will establish their own business
370 processes for it. For example, Service Provider may have a requirement that the mere Trust Network
371 membership is not sufficient, thus it needs to invoke the Service Provider specific Client Intake busi-
372 ness process to get the Client vetted and registered. If this happens in the background on the Service
373 Provider side, the main flow is not affected. If, however, the Client Intake business process needs to
374 be initiated by the Client, then the Service Provider needs to return an error code or policy requirement
375 that triggers the Client to initiate the process. In this case there needs to be a private understanding
376 about the meaning of these error codes between the Client and the Service Provider. TAS³ foresees
377 this mechanism, but does not specify what the codes are.

378 **6.1 Expressing Trust and Policy Pledges and Requirements**

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

- 381 A. What policies Client pledges to honour
- 382 B. What policies Client requires SP to honour
- 383 C. What policies SP pledges to honour
- 384 D. What policies SP requires Client to honour

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

389 **6.1.1 Expressing Pledges and Requirements in Discovery Phase**

390 Some trust and privacy requirements can be expressed as [CARML] declarations, permitting fairly
391 fine grained specification of the needs of the Client and the policies it is willing to respect. [AAPML]



6.1. EXPRESSING TRUST AND POLICY PLEDGES AND REQUIREMENTS

392 declaration allows a service to express its policy requirements so that Clients know what they need to
393 honour if they plan to communicate with the service. Both CARML and AAPML work best during the
394 discovery phase (and discovery registration).

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

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

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

404 6.1.2 Expressing Pledges and Requirements in Call Phase

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

408 The Client expresses its acceptance of the negotiation by performing a service request. If it has not
409 obtained an acceptable policy from the Sp, then it can not make the request. Currently (May 2009)
410 there is no mechanism for Client to request relaxation of the policy, other than just try request with
411 policy that is formally known to be unacceptable to the SP. More elegant solution to this situation is an
412 area of active TAS³ research.

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

416 Given the 4 requirements stated in beginning of this Section 6.1, the gap analysis against [SOAPBinding2]
417 indicates that (A) is well covered by <UsageDirective>. Specification of (B) is not expressly called
418 out, but the <UsageDirective> in the request can be used for this purpose as well. In call phase
419 there is no support for (C), but discovery phase supports it relatively well. Only problem is that the dis-
420 covery phase service pledge can not take the specific resource and operation in account. If this really
421 turns out to be a requirement, future versions of TAS³ architecture may specify some mechanism for
422 doing this. Support for (D) is two fold: the SP sees from request <UsageDirective> what the Client
423 pledging and can then include in response a <UsageDirective> to refine the requirements. If the SP
424 wishes to impose requirements that are not mere refinements, then it must refuse the service call and
425 provide a <UsageDirective> expressing what it would have accepted.

6 Bibliography

- 426
- 427 [AAPML] Prateek Mishra, ed.: "AAPML: Attribute Authority Policy Markup Lan-
428 guage", Working Draft 08, Nov. 28, 2006, Liberty Alliance / Oracle.
429 <http://www.oracle.com/technology/tech/standards/idm/igf/pdf/IGF-AAPML-spec-08.pdf>
- 430 [CARML] Phil Hunt and Prateek Mishra, eds.: "Liberty IGF Client Attribute Requirements
431 Markup Language (CARML) Specification", Draft 1.0-12, Liberty Alliance, 2008.
432 http://www.projectliberty.org/liberty/resource_center/specifications/igf_1_0_specs
- 433 [Disco2] Cahill, ed.: "Liberty ID-WSF Discovery service 2.0", liberty-idwsf-disco-svc-2.0-errata-
434 v1.0.pdf from http://projectliberty.org/resource_center/
- 435 [PeopleSvc] "Liberty ID-WSF People Service Specification", liberty-idwsf-people-service-1.0-errata-
436 v1.0.pdf from http://projectliberty.org/resource_center/specifications/
- 437 [RFC2560] Myers et al., "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol
438 - OCSP", RFC 2560, June 1999.
- 439 [SAML2core] "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML)
440 V2.0", Oasis Standard, 15.3.2005, saml-core-2.0-os
- 441 [SAML2meta] Cantor, Moreh, Phipott, Maler, eds., "Metadata for the OASIS Security Assertion
442 Markup Language (SAML) V2.0", Oasis Standard, 15.3.2005, saml-metadata-2.0-os
- 443 [SOAPAuthn2] "Liberty ID-WSF Authentication, Single Sign-On, and Identity Map-
444 ping Services Specification", liberty-idwsf-authn-svc-2.0-errata-v1.0.pdf from
445 http://projectliberty.org/resource_center/specifications/
- 446 [SOAPBinding2] "Liberty ID-WSF SOAP Binding Specification", liberty-idwsf-soap-binding-2.0-errata-
447 v1.0.pdf from http://projectliberty.org/resource_center/specifications
- 448 [TAS3ARCH] Sampo Kellomäki, ed.: "TAS3 Architecture", TAS3 Consortium, 2009. Document: tas3-
449 arch-vXX.pdf
- 450 [TAS3PROTO] Sampo Kellomäki, ed.: "TAS3 Protocols and Concrete Architecture", TAS3 Consor-
451 tium, 2009. Document: tas3-proto-vXX.pdf
- 452 [WSTrust] "WS-Trust 1.3", CD 6, OASIS, Sept 2006. (** WS-Trust, STS, etc.)
- 453 **Document ID** `tas3-disco-v04.pdf`
- 454 **URL path** `https://portal.tas3.eu/arch/review/tas3-deliv-4_1-disco-v04.pdf`