Electronic ID at work: issues and perspective

Antonio Lioy

< lioy @ polito.it >

Politecnico di Torino
Dip. Automatica e Informatica

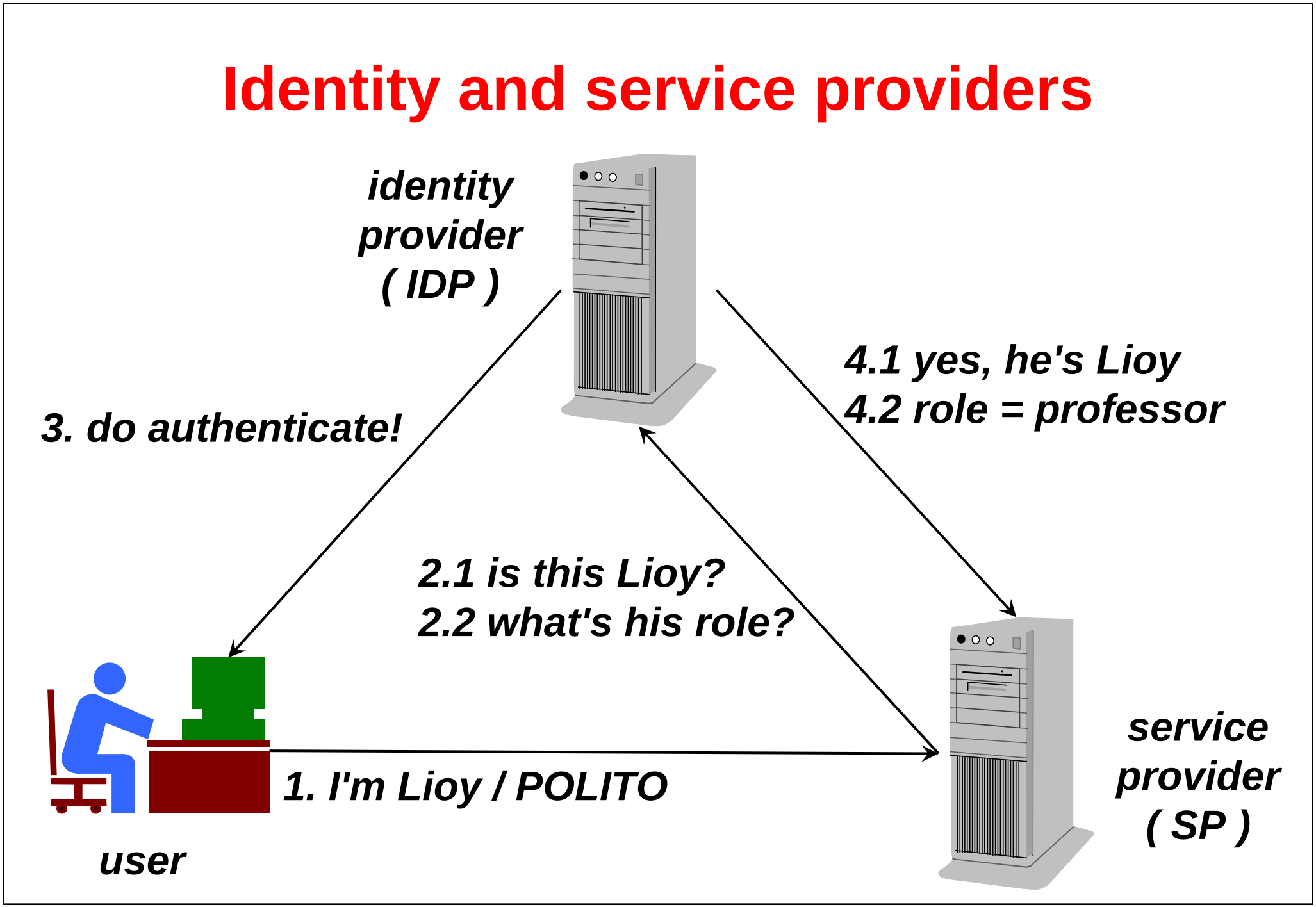
Why should I have/use an (e-) ID?

- **to prove my identity to an "authority":**
  - e.g. crossing borders
    - being an Italian citizen, I'm allowed to freely travel across EU and some foreign countries, while other countries require an entry visa
    - identity not really important, rather being Italian
- **to prove "ownership" of something:**
  - (access control) a credit card, a mail account, ...
    - non necessarily my "real" identity
  - (data origin) an e-document, a song, ...
- **to have my actions being tracked (!!!)**
  - e.g. in Italy no anonymous Internet access
  - difficult balance between privacy and lawful investigation

Electronic Identification

- **(peer) authentication:**
  - submit credential (e.g. username) to access control point
  - pass an associated verification (e.g. password)
- **(data) authentication:**
  - several electronic ways (e-signature, TTP, …)
  - not today's' topic
- **attributes:**
  - of the authenticated peer
  - some basic data openly available (e.g. name and surname)
  - other data available on explicit consent (e.g. religion)

Identity and service providers

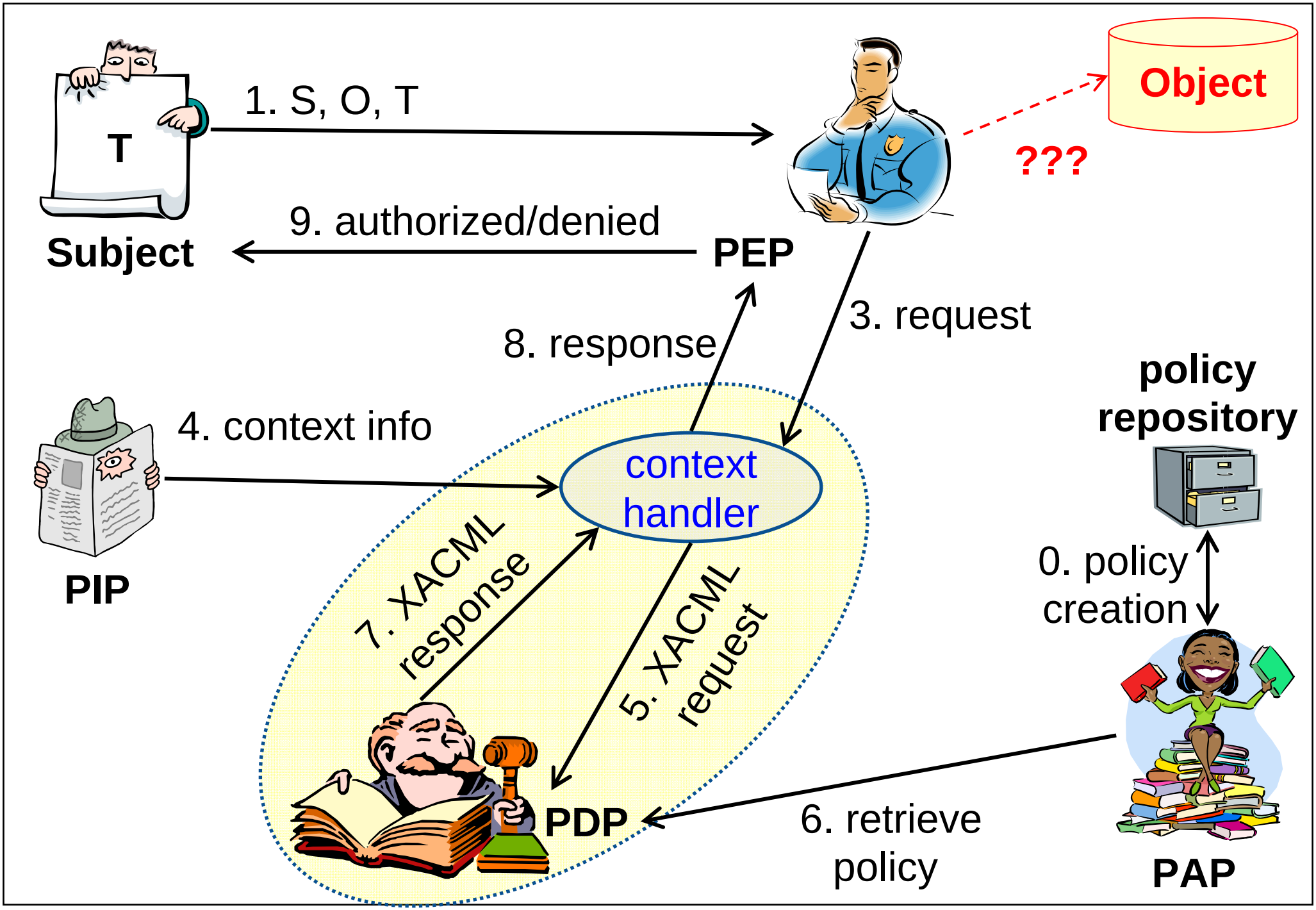


What is XACML?

- **eXtensible Access Control Markup Language**
- **an OASIS standard based on a XML syntax**
- **a language to describe an authorization policy, defined in terms of:**
  - subject (user, computer, service)
  - resource (document, file, data) identified by a URI
- **a language to manage policy-based access control:**
  - data format to express the request and response
  - to be carried inside one of many client-server protocols

Components for policy-based access control

- **PEP = Policy Enforcement Point**
  - front-end to a protected resource that permits access only after checking compatibility with access policy
- **PDP = Policy Decision Point**
  - collects all relevant information (policy, subject, resource, access type, context) to decide if access is allowed or denied
- **PIP = Policy Information Point**
  - provides auxiliary information related to the access request
- **PAP = Policy Access Point**
  - provides the relevant policy for the access request


T
1. S, O, T
Object
???
Subject
9. authorized/denied
PEP
3. request
8. response
policy
repository
4. context info
context
handler
PIP
7. XACML
response
5. XACML
request
0. policy
creation
PDP
6. retrieve
policy
PAP

Authentication

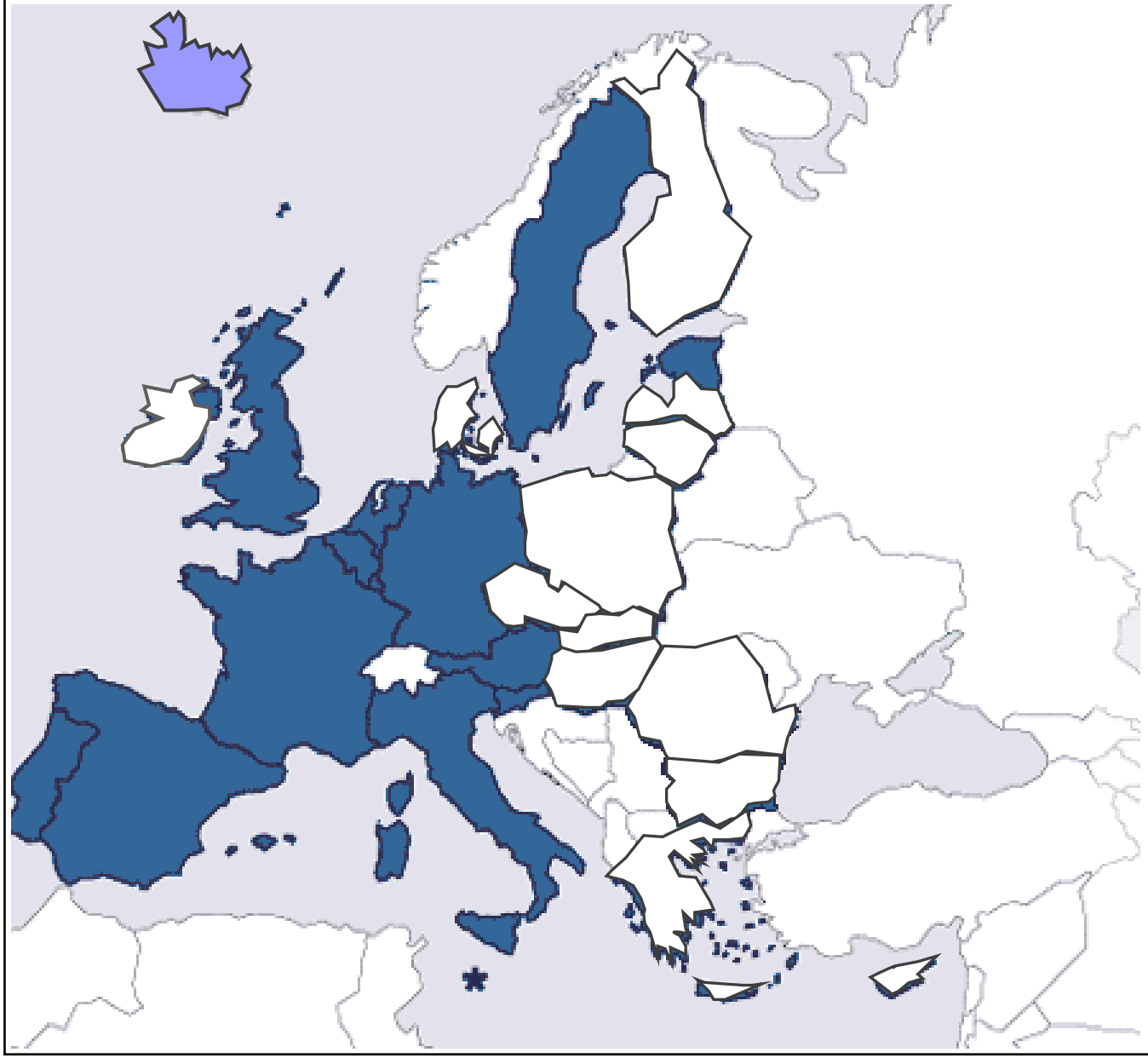
- **reusable password**
  - simple but highly risky
  - non-repudiation impossible
- **one-time password**
  - hw support needed but much less risky
  - non-repudiation impossible
- **asymmetric key (~ digital signature)**
  - smart-card needed (+pin) but highly secure
  - non-repudiation possible
  - with standard protocols (e.g. SSL client authentication available in all browsers)
  - with custom protocols (customized sw needed at the local node connected to the smart-card)

The STORK project

- **European PSP project (NOT a research project!)**
  - e-ID interoperability
- **purpose**
  - demonstrate the use of existing national e-ID for use with pan-European electronic services
- **funding**
  - 20 M Euro
- **duration**
  - June 2008 – June 2011
- **coordinator**
  - ATOS Origin



The STORK partners



- ✓ Austria
- ✓ Belgium
- ✓ Estonia
- ✓ France
- ✓ Germany
- ✓ Italy
- ✓ Luxembourg
- ✓ Netherlands
- ✓ Portugal
- ✓ Slovenia
- ✓ Spain
- ✓ Sweden
- ✓ United Kingdom
- ✓ Plus – Iceland

- ✓ addition of other countries under negotiation

STORK – pilot projects

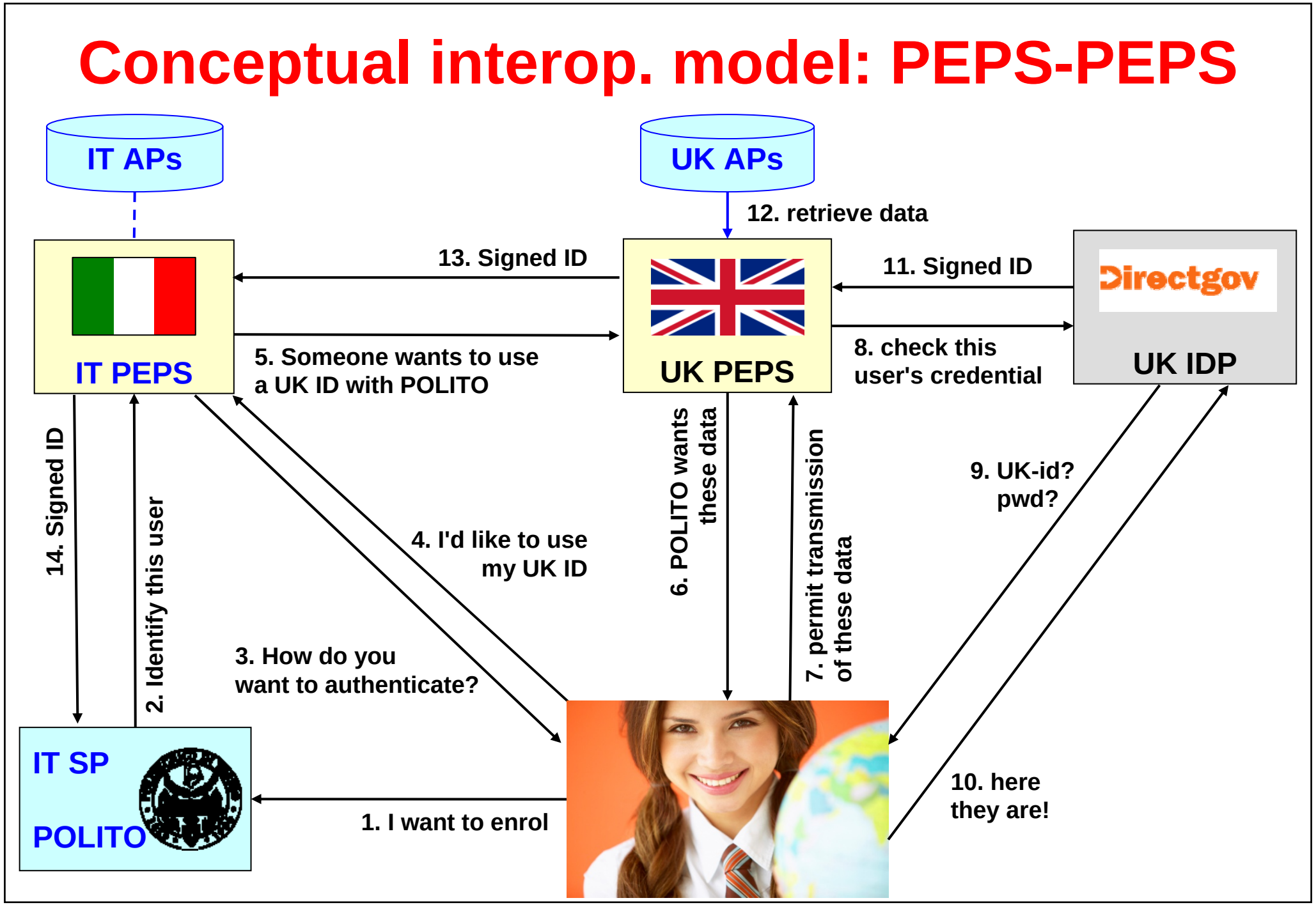
- P1) cross-border e-services between various national, regional portals
- P2) platform for safer chat for minors
- P3) electronic services to students attending an university abroad (e.g. ERASMUS)
- P4) cross-border secure online delivery of documents
- P5) change of address across EU countries

- common architecture for Internet-based services to allow re-use of national e-ID outside the origin country

Security in STORK

- **problems:**
  - nearly every EU country has a different e-ID schema
    - password, smart-card (w or w/o local software or portal)
    - different format / functionality of smart-cards
- **solutions:**
  - PEPS (Pan-European Proxy Service)
  - EU middleware (for some smart-card-based e-IDs)
  - "trust levels"
    - some services require a certain security level not provided by every e-ID

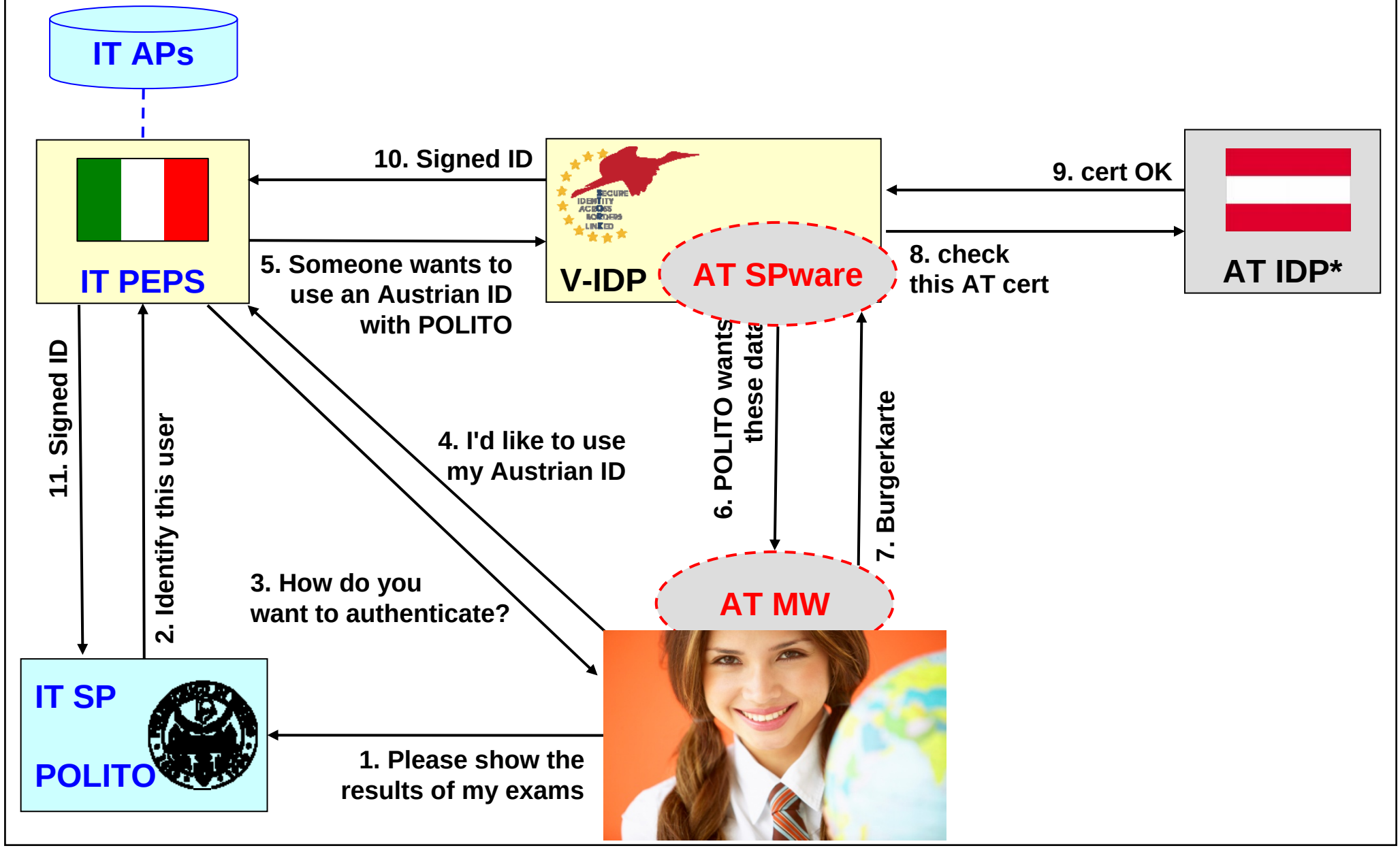
Conceptual interop. model: PEPS-PEPS



Conceptual interop. model: PEPS-PEPS

- **trust between PEPSes, PEPS-AP & PEPS-IDP required**
- **trust between PEPS - SP depend on model: UK vs BE:**
- **UK model:**
  - PEPS/DirectGov determines max attributes of each SP
  - effort for inclusion of new SPs
- **BE model:**
  - risk of DoS, as there’s no limitation on requests
  - no inclusion of SPs

Conceptual interop. model: PEPS-MW



Conceptual interop. model: PEPS-MW

- MW is activated by AT-SPware installed in V-IDP
- V-IDP also includes DE-SPware
- PEPS can collect more data items from AP, neither MW nor SPware will do so

What about standards?

- **standards**
- **... they are nice because you have so many to choose from!**
- **... and each standard has so many options to choose from!**
- **Stork will to exploit widely adopted standards**
  - SAML 2.0 is one of them

Conclusions

- **e-IDs are already here**
- **interoperability is possible**
- **user attributes (semantically meaningful) are the real challenge**

What is SAML?

- **Security Assertion Markup Language**
- **an OASIS standard based on a XML syntax**
- **a data format to express:**
  - several types of assertion
  - assertion requests
  - assertion responses
- **ASSERTION is the base SAML object**
- **the main purpose of SAML is to standardize and simplify the interactions needed to establish permissions in a multi-domain distributed system**

SAML assertion

- **an assertion is:**
  - a declaration about a fact related to a subject (e.g. the role of a user)
  - declaration provided by a certain issuer
- **three basic assertion types:**
  - authentication
  - attribute
  - di authorization decision
- **can be extended to add other assertion types**
- **the assertion may be digitally signed (via xml-dsig)**

Infos common to all SAML assertions

- **issuer and issuance timestamp**
- **assertion ID**
- **subject**
  - ID and its security domain
- **validity "conditions"**
  - SAML clients must reject assertions containing unknown conditions
  - an important condition: assertion validity period
- **other useful infos**
  - e.g. explanation / proof of the ground for the assertion

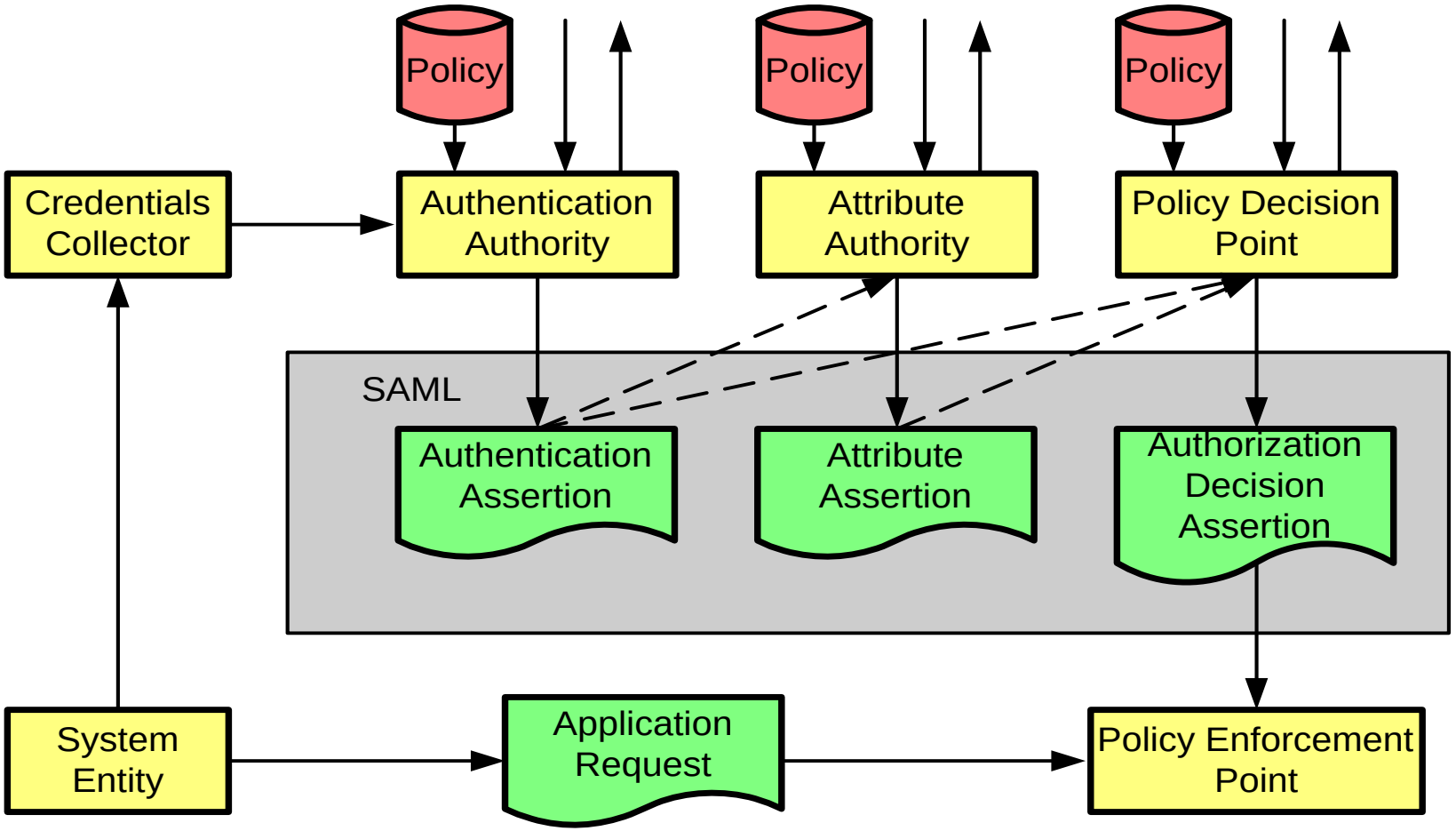
Example of authentication assertion

```
<saml:Assertion
     MajorVersion="1" MinorVersion="0"
     AssertionID="192.168.1.1.12345678"
     Issuer="Politecnico di Torino"
     IssueInstant="2007-12-03T10:02:00Z">
     <saml:Conditions
       NotBefore="2007-12-03T10:00:00Z"
       NotAfter="2007-12-03T10:05:00Z" />
     <saml:AuthenticationStatement
       AuthenticationMethod="password"
       AuthenticationInstant="2007-12-03T10:02:00Z">
       <saml:Subject>
         <saml:NameIdentifier
           SecurityDomain="polito.it"
           Name="alioy" />
       </saml:Subject>
     </saml:AuthenticationStatement>
   </saml:Assertion>
```

Example of attribute assertion

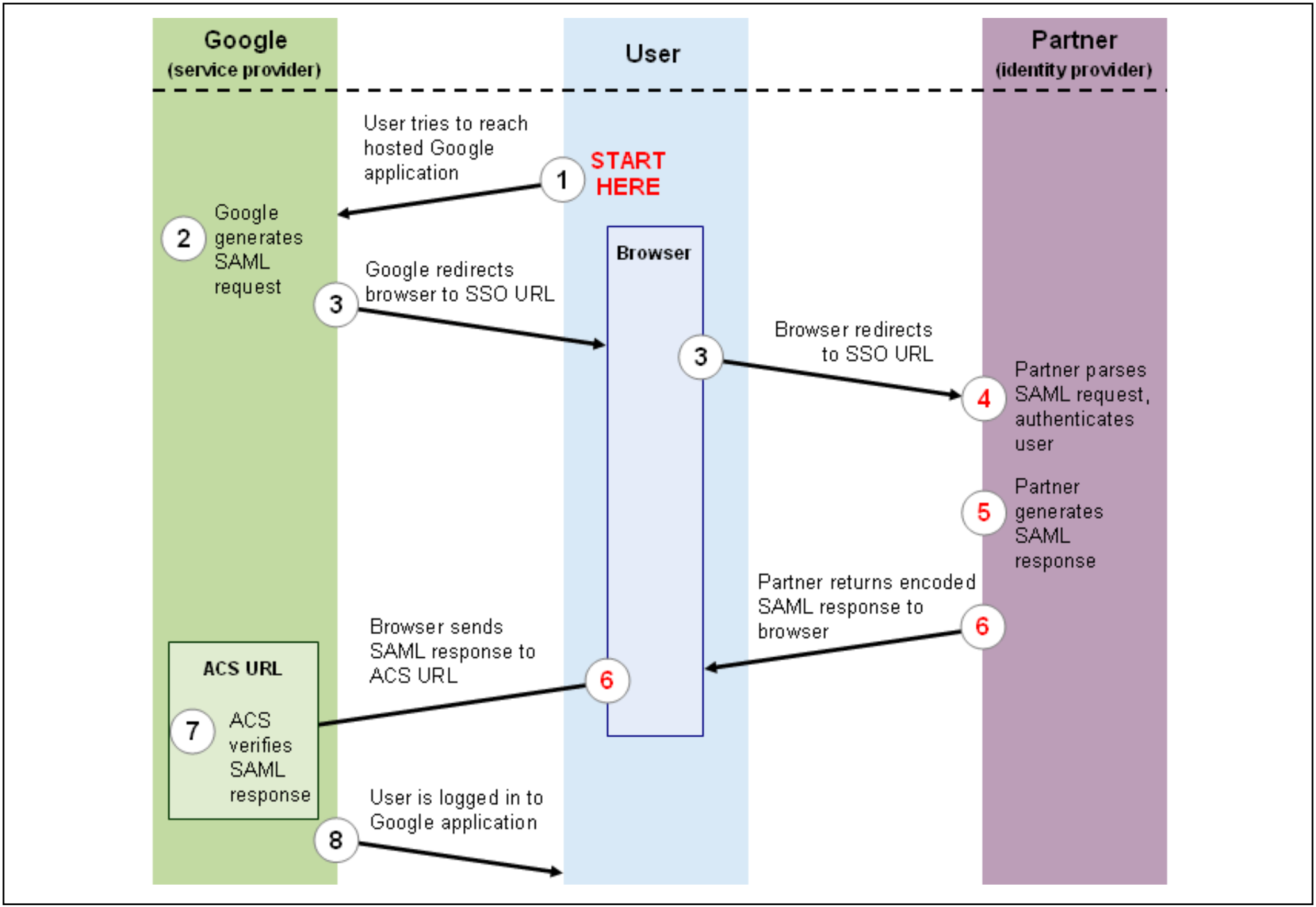
```
<saml:Assertion ...>
   <saml:Conditions .../>
   <saml:AttributeStatement>
      <saml:Subject>
         <saml:NameIdentifier
            SecurityDomain="polito.it"
            Name="alioy" />
      </saml:Subject>
      <saml:Attribute
         AttributeName="Role"
         AttributeNamespace="http://polito.it">
         <saml:AttributeValue>
            Full Professor
         </saml:AttributeValue>
      </saml:Attribute>
   </saml:AttributeStatement>
</saml:Assertion>
```

SAML: producer-consumer model



SAML SSO for Google Apps

- **a company (partner) activates its own application at Google**
  - hence, Google = service provider
- **the partner wants to keep control of the authentication and authorization part**
  - hence, partner = identity provider
- **the access procedure is based upon SAML-2.0 with XML-sig**


Google
(service provider)
User
Partner
(identity provider)
User tries to reach hosted Google application
1
START HERE
2
Google generates SAML request
Browser
Google redirects browser to SSO URL
3
3
Browser redirects to SSO URL
4
Partner parses SAML request, authenticates user
5
Partner generates SAML response
Partner returns encoded SAML response to browser
6
Browser sends SAML response to ACS URL
6
ACS URL
7
ACS verifies SAML response
User is logged in to Google application
8