Open Philosophies for Associative Autopoietic Digital Ecosystems

Contract n° 034824

Workpackage 4: Distributed Accountability, Identity, and Trust

Deliverable D4.1: Distributed Identity Model for the Digital Ecosystem

Project funded by the European Community under the “Information Society Technology” Programme

Contract Number: 034824

Project Acronym: OPAALS

Title: Open Philosophies for Associative Autopoietic Digital Ecosystems

Deliverable N°: D4.1

Due dates: 05/2007

Delivery Date: 08/2008

Short Description:

The purpose of this document is to provide a distributed Identity Management Model for digital ecosystems.

The document includes an introduction to identity and technological approaches to digital identity. A distributed identity model is described in detail. Implementation decisions and approach is provided as well as details of integrating this approach with the trust overlay approach introduced in D4.3

We conclude with a summary of the work and an indication of future work required in this field

Author: WIT, TI, CN, LSE

Partners contributed: WIT

Made available to: Public

Versioning		
Version	Date	Author, Organisation
1.0	**05/2007**	**Javier Noguera, TechIdeas**
2.0	**11/2007**	**Mihaela Ion, CreateNet**
2.1	**25/11/07**	**Paul Malone, Mark McLaughlin, WIT**
2.2	**03/02/08**	**Paul Malone, Mark McLaughlin, WIT**
2.3	**01/08/08**	**Mary Darking, Panayiota Tsatsou, LSE**
2.4	**14/08/08**	**Paul Malone, Mark McLaughlin, WIT**

Quality check:

1st Internal Reviewer :

2nd Internal Reviewer:

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 License. To view a copy of this license, visit : http://creativecommons.org/licenses/by-nc-sa/3.0/or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

Table of Contents

1 Introduction....6
2 Identity and Digital Identity....7
2.1 Social Aspects....7
2.2 Real World Identity....7
2.3 Basic identification standards....8
2.3.1 X.509....8
2.3.2 SPKI....8
2.4 User Centric Identity Management Technological Standards....8
2.4.1 YADIS....8
2.4.2 OpenID....9
2.4.3 Light-Weight Identity (LID)....9
2.4.4 XRI/XDI....9
2.4.5 iNames....9
2.4.6 InfoCard....10
2.4.7 Higgins trust framework....10
2.4.8 SXIP 2.0, (Simple, eXtensible Identity Protocol)....11
2.4.9 DIX....11
2.5 Distributed Identity Management Standards....11
2.5.1 SAML....11
2.5.2 Liberty Alliance....13
2.5.4 WS-Trust and WS-Federation....13
3 Identity Model for Digital Ecosystems....14
3.1 Multiple user identities....15
3.2 User profile....16
3.3 Passwords and secure tokens....17
3.4 Use of SAML in the model....18
3.5 Model communication scheme....19
3.6 Model extension to identity propagation in service composition....22
4 Model Implementation....25
4.1 Technologies used and design decisions....25
4.2 Refining the model....26
4.3 Actors and Connections....27

4.4 Building Operations from Profiles....27
4.5 Actors and State....30
4.6 Open Source Project - IdentityFlow....30
4.7 Integrating Identity and Trust Models....32
5 Concluding Remarks and Future Directions....34

Illustration Index

Figure 1: The i-Name Concept (source: http://www.inames.net)....10
Figure 2: SP-Initiated SSO with Redirect and POST Bindings....13
Figure 3: Model communication scheme....20
Figure 4: Service Composition using Proxy Certificates....23
Figure 5: Implementation using a 'pure SAML' approach....26
Figure 6.: Single Sign-On Operation based on the SAML SSO profile....28
Figure 7: Single Sign-On Operation (with simplified view OpenID authentication)....29
Figure 8: The IdentityFlow Project Web Site....31
Figure 9: Distributed Identity Model Integrated with Trust Overlay Model....33

1 Introduction

WP4 of OPAALS is concerned with defining models for distributed identity, trust and accountability in digital ecosystems. Identity is the foundation of these combined concepts in that it provides concrete links to entities whose actions can to be accounted for and in which trust can be placed. The notions of identity and reputation constitute key elements of trust both in social science and computer science. From a social science perspective, reputation based on the identity of agents enables trust relationships between agents, encouraging interaction and exchange. From a processual perspective, identity is to be understood as highly dependent on the context and trust relationships required for communication and elaboration between two or more participants in the system. In other words, depending on the trust criteria that someone uses to evaluate his or her collaborator, the identity elements assigned to the latter vary.

The technical challenges in defining a distributed identity solution lie primarily in the requirement that no single point of failure exists. Traditional digital identity solutions (e.g. digital certification) rely on the existence of a centralised authority in the validation of identities. Our approach was to define a technology agnostic model which allowed for the assertion of identities through a distributed trust infrastructure, without the need for a centralised authority and also capable of incorporating legacy identity solutions which ecosystem entrants may already have in place in their infrastructure.

The remainder of this document is structured as follows. Section 2 provides an overview of identity from a technological viewpoint by discussing approaches and standards appropriate for our discussion. Section 3 describes the initial distributed identity model which incorporates a composed service solution for identity propagation. Section 4 describes some refinements to this model and also implementation decisions and a description of an open source project (*IdentityFlow*) where this implementation is being maintained. This section also provides a brief description of how this identity model is being integrated with the distributed trust model as described in Deliverable D4.3. The document concludes with a brief section describing the work performed and the future work which will be undertaken in the next stages of the project.

2 Identity and Digital Identity

This section mainly describes technological approaches to and standards for digital identity. We start with a brief view of what identity means from a social science perspective and a description od how identity is linked to trustworthiness in the real world. Then follows a description of basic digital identity approaches and a look at user centric identity efforts as well as a brief description of some standards for distributed identity.

2.1 Social Aspects

Identity in social science is often understood as the attributes of 'self' and is variously defined in disciplines such as psychology, sociology and social anthropology. Typical example is Erik Erikson's psychological framework of identity and the distinction between the ego identity (the psychological sense of self), the personal identity (the personal traits that separate one person from another) and the social or cultural identity (social roles that a person might play). For Erikson, the development of a strong ego identity and the proper integration into a society and culture may lead the person to a stronger sense of identity. Accordingly, a deficiency in any of these three parameters may lead to an identity crisis.

In electronic commerce, authentication is the process whereby an entity (person, computer or organization) establishes that another entity is who it claims to be. Thus, authentication involves issues such as membership, access rights, and public certification of identity. Authentication can facilitate trust relationships since it provides the means for identifying any malpractice of the system participants. Digital signatures are one of the most widespread methods of proving identity to those who do business online.

2.2 Real World Identity

In the non-digital world we use various forms of identity on a daily basis. For example passports, work identity cards, drivers licences, credit cards all perform as a means of asserting identity and in turn grant access rights to information, rights of passage and financial transactions. All of these are predicated upon a trust in the issuers of these identity tokens. For example a passport is accepted as

a means of identifying an individual only when the third party has trust in the issuer of the passport (e.g.. Immigration control in the USA trusts the Irish Department of Foreign Affairs to issue an accurate passport document when asserting the identity of an Irish citizen who wishes to enter the USA).

2.3 Basic identification standards

2.3.1 X.509

X.509 [1]is the widely used standard for Public Key Infrastructure (PKI). It specifies a standard format for certificates and a certification path validation algorithm. An X.509 certificate binds a public key to a global name. A Certification Authority (CA) is a trusted third party which creates and signs off-line digital certificates. X.509 assumes a hierarchy of CAs. X.509 was designed for providing authentication and attribute information.

2.3.2 SPKI

Simple Public Key Infrastructure(SPKI) [2] provides a standard for using digital certificates to provide authorisation and authentication for using resources in a peer-to-peer fashion. SPKI allows principals to define local names and their namespaces can be linked to define trust (also called web-of-trust). There is no hierarchical global infrastructure in contrast to X.509. Each public key (entity) is a certificate authority and can issue certificates on the same basis as any other principal thus making it suitable for decentralised and unstable coalitions.

2.4 User Centric Identity Management Technological Standards

2.4.1 YADIS

YADIS[3] is an open initiative to build an interoperable lightweight discovery protocol for decentralised, user-centric digital identity and related purposes. With Yadis, the capabilities of identities can composed from a open-ended set of services, defined and/or implemented by many different parties.
By allowing each party in an online relationship to choose the authentication and data sharing protocols they want to use to share their information, Yadis fosters the development of mutual trust and respect. Yadis aims to make the internet a more people-friendly place by putting the human

being at the centre of their own online experience, letting them define what information they expose and which services they use

2.4.2 OpenID

OpenID[4] is a simple identification mechanism advanced by Brad Fitzpatrick of LiveJournal. It is a distributed, decentralised network, in which your identity is a URL, and can be verified by any server running the protocol. It is part of the YADIS family of protocols. The philosophy is different from single sign-on, where authentication plays a major part, because it does not rely on a trust mechanism. Therefore, OpenID is not meant to be used on sensitive accounts (banking, on-line purchasing and so on).

2.4.3 Light-Weight Identity (LID)

Light-Weight Identity (LID) [5] is a set of protocols and software implementations created by NetMesh Inc. for representing and using digital identities on the Internet without relying on any central authority. LID supports digital identities for humans, human organisations and non-humans (e.g. software agents, things, websites, etc.) It implements Yadis, the identity and accountability foundation for Web 2.0.

Unlike other digital identity systems, LID is organised in a base protocol called MinimumLID, and an ever-growing list of profiles on top of it. This enables LID to be a foundation for digital-identity related innovation by many parties. Any implementer chooses which or how many LID profiles to support to meet their needs.

2.4.4 XRI/XDI

XRI/XDI (Extensible Resource Identifier/XRI Data Interchange) [6][7] is maintained by XDI.ORG is an international non-profit organisation governing public services based on the XRI abstract identifier and XDI data interchange protocols under development at OASIS. This new layer of infrastructure enables individuals and organisations to establish persistent Internet identities and form long-term, trusted peer-to-peer data sharing relationships.

2.4.5 iNames

iNames[8] are one form of an XRI- an OASIS open standard for abstract identifiers designed for sharing resources and data across domains and applications. One problem XRIs are designed to solve is *persistent addressing* - how to maintain an address that does not need to change no matter

how often the contact data for a person or organisation changes. XRIs accomplish this by adding a new layer of abstract addressing over the existing IP numbering and DNS naming layers used on the Internet today. Privacy is protected because the identity owner controls this resolution.

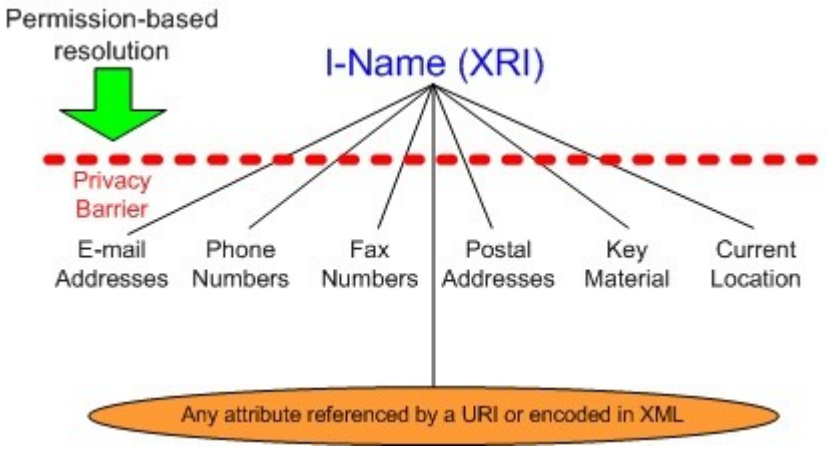


Figure 1: The i-Name Concept (source: http://www.inames.net)

2.4.6 InfoCard

InfoCard (also known as CardSpace) [9] is a software component which securely stores digital identities of a person, and provides an unified interface for choosing the identity for a particular transaction, such as logging in to a website. InfoCard is a central part of the Microsoft effort to create an *Identity Metasystem*, or an unified, secure and interoperable identity layer for the Internet. The InfoCard software allows the users to create *self-signed* identities for themselves.

InfoCard is built on top of Web Services Protocol Stack, an arguably open set of technologies, including WS-Trust. InfoCard competes with other Internet identity architectures like YADIS and Liberty Alliance, although in some ways the three standards can be seen as complementary.

2.4.7 Higgins trust framework

Higgins trust framework [10]is a set of open source protocols and software applications that allow people to store their digital identities on their personal computers and share the stored information with commercial companies and other parties in a controlled fashion. Higgins is sponsored by IBM and Novell and is promoted as an alternative to Microsoft InfoCard. Higgins used to be called Eclipse trust framework, and is a project of the Eclipse Foundation

2.4.8 SXIP 2.0, (Simple, eXtensible Identity Protocol)

The latest version of the Simple eXtensible Identity Protocol, SXIP 2.0 [11] is a protocol for automating the exchange of identity data on the Internet. SXIP 2.0 has a new improved and decentralised architecture.

Single Sign-On – access to different websites

User-Centric Verified Identity - allows users to acquire and release verified "assertions" around their identity, enabling them to create richer profiles of their online identity.

User Choice – supplies added privacy by enabling users to be actively involved in the release of the data they store in their identity profile.

2.4.9 DIX

DIX (Digital Identity Exchange) [12] is an Internet scale protocol for exchanging identity information between endpoints. The protocol architecture maintains a separation of control between all parties of the exchange and supports both compartmentalised and anonymous identities. The end-user should have control over their identity information and always be providing consent for the exchange of any of their information. Initiation of an exchange between two parties should be possible without any pre-established relationship.

2.5 Distributed Identity Management Standards

There are several technologies and standards used for managing distributed identities. The most mature and widely deployed solutions for federated identity are the Security Assertion Markup Language (SAML) and Liberty Alliance standards.

2.5.1 SAML

SAML [13], developed by OASIS, is an XML-based framework for communicating user authentication, authorisation and attribute information. SAML provides XML formats and protocols for encoding and exchanging identity information. It also provides standards for single sign-on of identity management.

SAML assertions allow principals to make statements about a subject's authentication, attribute, or authorisation details. A subject is uniquely referred to by using an *Identifier* which can be a real

name or a pseudonym. SAML focuses on authentication and attribute statements while authorisation statements are the focus of XACML [14] . The entity issuing the SAML assertions is called the asserting party or *identity provider* (IdP). The party which makes use of the assertions made by the IdP to control access and provide services is called relaying party or *service provider* (SP). A SP trusts an IdP if it relies on assertions issued by the IdP.

Identity federation means sharing of identity information between domains who have a trust relationship or agreement. SAML and Liberty Alliance define standards for federating identities and single sign-on (SSO). SSO allows users to get identified once and then move across domains within a federation with this identity.

The SSO use case requires a Service Provider (SP) and an Identity Provider (IdP) which have a trust relationship (SP relies on assertions from IdP). The SSO can be initiated by any of the two parties. Below we describe the SP-initiated SSO use case. The steps are also illustrated in Figure 2 [15].

1. The user tries to access a service on SP without any login information. (step 1)
2. The user is not recognised and gets redirected to the trusted IdP. (step 2)
3. The user signs on by providing the required credentials to the IdP (e.g. username and password). (steps 3 and 4)
4. IdP provides a SAML web SSO assertion for the user's federated identity back to SP. (steps 5 and 6)
5. The user is identified at SP and can access the service. (step 7) The redirection is done automatically and the user is not aware of being forwarded to a different domain.

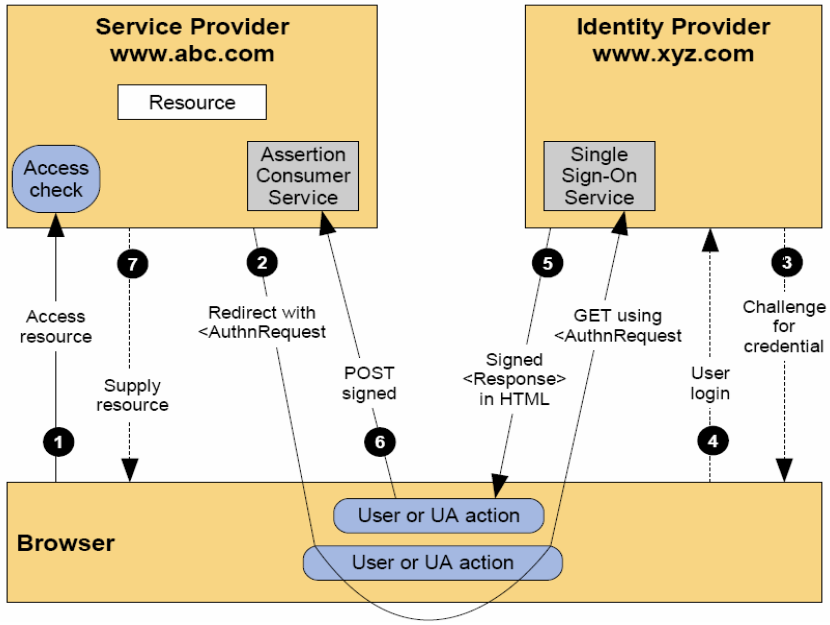


Figure 2: SP-Initiated SSO with Redirect and POST Bindings

2.5.2 Liberty Alliance

Liberty Alliance provides open SAML based standards for federated network identity. The most relevant technology specifications developed by the Alliance are Identity Federation Framework (ID-FF) [16] and Web Services Framework (ID-WSF)[17]. As of the new SAML version (v2.0) the OASIS technical committee has unified the Liberty standards within one SAML identity framework with a rich set of identity profiles.

2.5.3 WS-Trust and WS-Federation

WS-Trust [18] and WS-Federation [19] define standards for federating identities by allowing and brokering trust of identities, attributes and authentication between participating Web services. However, though partially inspired by the two standards, our model aims at simplifying them and targeting environments composed of unstable coalitions which is often the case for SMEs.

3 Identity Model for Digital Ecosystems

The model we present in this deliverable has been published by us in [20].

A Digital Ecosystem (DE) consists of diverse organisations which sometimes compete against each other and other times collaborate and form stable or unstable federations. Digital ecosystems are interconnected by a network to form a complex and dynamic environment.

Managing identities in such a distributed system poses many challenges. First of all, organisations use different types of certificates and identity technologies (e.g. X.509, SPKI and Kerberos) which are not always compatible with each other. Secondly, users often need to access applications, services or a composition of services located on different administrative domains. Finally, because of the dynamic nature of the environment, federating and sharing of identities becomes a complex task. A pure federating approach is viable only when there is a stable relation. In Digital Ecosystems, federation does not scale up because of the unstable and ad-hoc coalitions.

Companies co-operate some times and are rivals at other times. We need to provide ways for exchanging identity information between companies independent of the standards they use and to share user identity between different domains which could be federated or have no direct trust. WS-Trust, WS-Federation and WS-Policy [21] cover a wide range of requirements but are difficult to suit directly and simply for small and medium sized enterprises (SMEs). What SMEs in DEs need is a targeted model that is easy to understand and straightforward to implement and put in practice. Existing standards are heavy and difficult to understand and implement and therefore suitable mainly for large enterprises.

Our model aims at automating the process of identification between ecosystem partners. We focus on practical solutions which are clear and easy to implement. The model is based on the new SAML (v2.0) standard for providing proper identification. SAML provides interoperability on the message level and helps to automate and converge when the technologies are not compatible. We propose managing distributed identity storage by the use of user profiles. A user profile is an abstract view of a client's identity information that is stored in a decentralised manner in the peer-to-peer network.

We refer to the following key entities in our model:

- **User**: any entity that can be identified in the network (peer or web browser user, institution or person)
- **Service Provider (SP)**: any identifiable entity that has one or more services or resources available to other entities.
- **Credential Provider (CP)**: any entity that is able to provide digitally signed credentials to other entities.
- **Digital Ecosystem (DE)**: distributed digital environment where both partners and competitors are present and where stable and unstable coalitions are created; coalition of digitally represented partners with little (or no) previously established trust relations. Thus the notion of ecosystem comprises co-operative and competitive relations.
- **Federation**: stable coalition of companies which have a co-operative relation.

We propose an identity management model for decentralised peer-to-peer ecosystem domains. All users are considered equal and there is no hierarchy of ecosystems. Any peer can be a Credential Provider or a Service Provider, or both. Each user can issue a certificate to other users. Each user has a list of trusted Credential Providers. Each Credential Provider has a list of acceptable security tokens. A Credential Provider issues certificates to users either (i) based on secure tokens issued by the provider itself or (ii) based on trusted secure tokens (from Credential Providers with whom it has trust relationships) or (iii) based on user registration information.

3.1 Multiple user identities

In a system of interconnected digital ecosystems, users and companies use different kinds of certificates obtained from outside the system. Companies have their own X.509 certificates issued by Certification Authorities outside the system and which they are obliged by law to use when doing online transactions. SMEs often have their own proprietary solutions for identification of their employees such as username and password, ad hoc secure tokens or adoption of OpenID for Web-based access.

To approach proper identity management, first we need to define a way to cope with the incompatibility of the variety of standards and solutions. Here we borrow the concept of credential

transformation from one type to another as already introduced in the WS-Trust standard. To address the problem, we have to convert identity information from one certificate technology to another one compatible with the current domain of business.

After joining a Digital Ecosystem, users (partners) obtain a variety of certificate tokens issued (transformed) by Credential Providers for particular business needs. Possible secure tokens considered in the system are X.509, SPKI, Kerberos and SAML identity assertions. However, partners that already have ad hoc identity tokens (or username/password) can use them in the system but only for the purpose of providing identity information to CPs that are to certify partners' identity. All the subsequent certificates issued by CPs in a DE are bound to one of the standards mentioned above. The reason for that is to unify and simplify identity management between DEs to well-defined identity standards.

Each CP has the responsibility to provide proper pseudonimity to end users. Typically, a CP either provides a user pseudonym on its own or allows users to define it and then certifies the pseudonym in a trusted secure token to a Service Provider. We note that a SP explicitly asks a CP to reveal user identity in case of user misbehaviour. So, each CP maintains a database mapping user's pseudonimity with user's real identity.

3.2 User profile

Having multiple identity certificates issued by different CPs, it becomes difficult for a user to manage and allocate all of them when needed to access a service, especially in the case of distributed services. Users connect to a DE either via a portal (a Web browser) or via a rich client system installed on their computers. In either of the cases, a user needs a way to manage its credentials, username/passwords and public/private key pairs. For that purpose we adopted the use of *user profile*. A user profile contains all available information about user's identity obtained from the user's interactions within DEs. Its main purpose is to provide an abstract view of which identity credentials are available, where they are available (e.g. local or remote storage) and how to obtain them (e.g. via authentication to a CP by username/password or via LDAP[1] storage etc.).

Here, an important issue is how to allocate, store and retrieve the user profile. The profile contains sensitive information that is necessary when communicating with entities in a DE. So, the profile must be protected from unauthorised disclosure (no one except the owner of the file) and at the

[1]http://www.openldap.org/

same time must be available on demand (avoid denial of service/availability). To address these issues we opted for keeping the profile encrypted and replicated on trusted peers. The encrypted profile is only meaningful to its owner and reliably obtained via a trusted peer-to-peer network.

Another issue worth mentioning is the availability of a profile to be shared (used) by multiple entities. This may often be the case for SMEs where selected employees are allowed to use the profile and therefore represent the company in on-line business negotiations. So, we decided to provide a sticky policy with each profile that encodes who can use the profile and under what conditions. The sticky policy is optional and if not explicitly specified it has the default value of read and write permissions only for the profile's owner. A good solution for a sticky policy model is the use of Access Control Lists (ACL).

When a user starts a new session his profile is downloaded on a secure memory (e.g. browser sandbox) in its Web browser or local client and then decrypted in the memory. Once decrypted the profile is ready to be used and processed by the Web browser client or the local client. At the end of the session, the user's profile is encrypted and updated on the associated trusted peers.

In the case of a local client installed on user's own machine, the profile could be locally copied and stored so that it could be loaded from the client's machine next time. However, in this case the profile must also be stored and replicated on the other trusted peers in order to provide availability and actualisation if shared among multiple users.

3.3 Passwords and secure tokens

Each user has a pair of private/public keys used in all certificates issued by different authorities. For the sake of simplicity of the framework we assume one key pair. However, multiple key pairs are also possible providing that there is a proper mapping between available certificates and used key pairs in the user profile. User identity information is stored in a user profile that is encrypted and replicated on trusted peers. The user profile contains information about available certificates, public/secret key pair and user authentication information needed to access and obtain secure tokens.

User identity information obtained outside DEs should be updated in the user profile so that it can be used when the user does business interactions with partners within DEs. Especially, when a user

first registers to a DE and creates its initial profile, it decides whether to import the already available identity information. However, a user can start from no identity information and collect it on a step-by-step basis when interacting with Service Providers (and their CPs).

After each interaction with a CP, the user's client (web or local) automatically records the information on the new identity token for subsequent use. The information stored depends on the settings the user specified and the CP's policy. For example, an identity token may only be issued to be presented to a SP without being stored on the client profile. In this case, we record only the information on how to obtain a new token (e.g., by presenting another token or by username/password). In other cases, the identity token could be stored on a secure LDAP server trusted by the CP who issued the token so that it can be automatically obtained by a user via authentication to the server. In any case, after each interaction with a CP, the client profile stores the necessary information on what identity, what validity and how to obtain such.

The last issue left to be examined is how to keep user profiles encrypted. A user profile is encrypted with a long master password (usually key phrase) that is never stored and known only to the user. Thus, a user has to remember one login information and one master password in order to facilitate best security when using a DE.

3.4 Use of SAML in the model

Having designed the identity model, we faced the problem of incompatibility of different identity standards. X.509 and SPKI, the certificate standards most widely cited in the literature, differ in design. We have to provide ways for a client identified with one standard to be able to use his identity information when communicating with a SP using another standard.

Another issue we need to take into account was that SMEs often adopt their own (ad hoc) certificate tokens or different identity mechanisms (such as OpenID) to manage identities of their employees.

To cope with this wide range of identity mechanisms we have to make the following assumption. Each SP adopts the identity standard best suiting its needs but its related CPs should support by default the SAML standard (especially v2.0). It means that any SME could preserve its existing identity management infrastructure but should enhance its trusted CP with the ability to understand SAML. Furthermore, each CP must be able to issue SAML assertions derived from any of the standards the CP supports. Refer to the example given below.

With the new version of SAML, the standard allows to express identity information (in SAML assertions) within a context of any type of authentication (e.g. X.509, SPKI, Kerberos tickets, username/password etc.). Thus, the model uses SAML assertions to bridge different identification information and standards. Depending on the standards used by their SPs, CPs should be able to support different assertions such as:

- X.509 --> SAML assertion
- SPKI --> SAML assertion
- Kerberos --> SAML assertion
- Username & password -> SAML assertion

SAML assertions are used when accessing or negotiating with different ecosystem domains. Once we unify the identity and authorisation representations between CPs we can accommodate any identity model/requirements particular to a service provider.

Example: X.509 and SPKI identity exchange

Let us suppose that SP_1 only accepts X.509-based authentication to identify entities and that SP_1 trusts CP_1 to validate X.509 tokens. Now, if a user has a SPKI certificate issued by CP_2 that has a trust relationships with CP_1, then the user is able to identify itself to SP_1 by use of our model.

To do so, the user has to contact his CP_2 and request for a SPKI SAML assertion in order to identify itself to CP_1. Since CP_1 has trust in CP_2 for proper entity identification, CP_1 accepts the SAML assertion . SP_1 trusts CP_1 for identifying entities and provides access to the desired service.

3.5 Model communication scheme

So far, we have presented all we need to state our identity management model. Figure 3 shows the basic model and workflow of messages between the main actors.

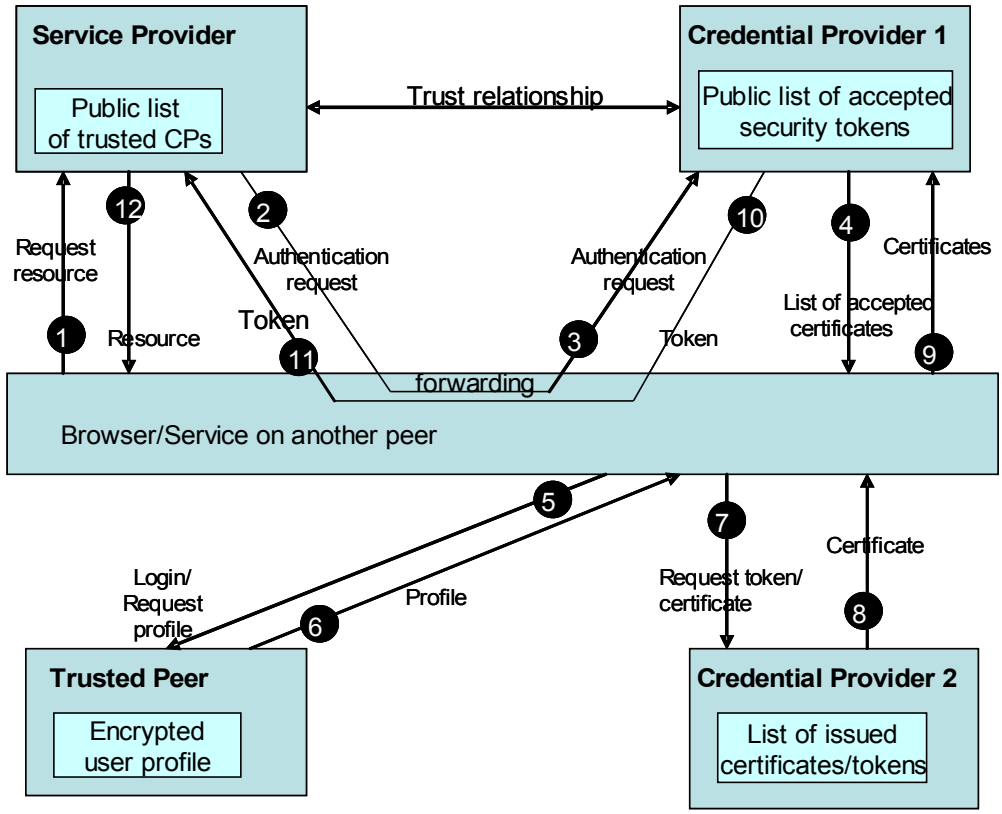


Figure 3: Model communication scheme

The message flow of the model is the following:

- 1. An entity (web browser user or local client) makes a request to a Service Provider.
- 2-3. The Service Provider redirects the user to a trusted Credential Provider (CP1).
- 4. The user has no credentials issued by CP1. CP1 sends a list of accepted certificates with a list of its trusted CPs to the user.
- 5. The user requests its profile from a trusted peer storing it and uses username/password for authentication. Information for ecosystem trusted peers is obtained (possibly publicly available) when users join the ecosystem.
- 6. The trusted peer sends the encrypted user profile.
- 7. After the profile is decrypted, the user checks if it has the right credentials, i.e. processes its profile for matching of credentials (issued by any of the CPs obtained in step 4). If no credential is matched, then the user has to register to CP1 to obtain an identity token. If one of the credentials requested in step 4 is found, then the user extracts it either from the profile

or requests it from the remote CP2 that issued it. Important issue here is that the user identifies whether the certificates match by type and CP name or only by CP name. The first case requires that the user just presents the certificate as it is, while the latter case requires that the user requests the CP for credential transformation.

- 8. CP2 authenticates the user and then returns either the requested certificate or its transformation to a SAML assertion.
- 9. The user forwards the SAML assertion to the CP1.
- 10. CP1 verifies the assertion and issues a new one that is to be forwarded to the SP.
- 11. The user is redirected to the Service Provider which accepts assertions from its CP1.
- 12. The Service Provider verifies the new certificate and provides the requested resource to the user.

The only case in which CP1 does not issue a new token is, for example, when the user has been already in contact with the SP and presents the same identity token that has been issued by CP1 last time. In this case, CP1 does only certificate verification and validation.

3.6 Model extension to identity propagation in service composition

Digital Ecosystems allow companies to co-operate with each other and compose services. An important requirement for an identity management model for DEs is to support composition of services. We extend the basic model presented above to cope with the case in which one service relies on services from other providers.

In some service composition scenarios, the service provider aggregating services from other service providers may need to run the services on the name of the user and as so he has to authenticate the user to the other providers. To solve this problem we adopted the use of *Proxy Certificate* [22] that the client issues to the provider of the composite service.

A Proxy Certificates is derived from and signed by a normal X.509 public key end entity certificate or by another Proxy Certificate (PC). The identity of the new PC is derived from the identity that signed it. A PC has its own public and private key pair. A PC is identified as such by its extensions. Any X.509 certificate has extension fields to encode different certificate characteristics. A PC has a policy that specifies what conditions must be respected when an entity is using it. Another important issue is that a PC can only sign another proxy certificate.

There are three important requirements specified in the policy of a proxy certificate that reflect our identity model. The first requirement is the scope of a PC. We identity the scope of a PC to be the scope of the service being requested by a client. Scope of service means any aggregated service that is directly used for the sake of proper execution of the main service. In other words, any service that is not directly aggregated by the main service (e.g. aggregation of aggregation) should not consider the PC as a valid identity certificate (on behalf of a client).

To solve the issue of complex aggregation of services that aggregate other services, we use a second policy: the level of aggregation. The purpose of this level of aggregation is to restrict the use of a PC in a chain of service aggregations.

The third policy requirement is the validity period of the PC. Usually, this depends on the particularity of the main service being executed (i.e., the validity of the service transaction). The client obtains such information from the SP hosting the main service.

The level of aggregation should be interpreted as not to use the PC as deep as the level is, but to indicate whether a new PC could be obtained from the original one. That is, when a SP contacts another SP to execute an aggregated service, the second SP specifies that it needs a PC to execute other services within its aggregated service. To do so, the first SP signs a new PC to the second SP but with level of aggregation decreased with one unit and scope of PC the scope of the second SP aggregated service. Additionally, the validity period of the new PC is the remaining validity period of the PC that signs it.

Thus, the second SP can use the new PC only for the sake of execution of its aggregated service as requested by the first SP and within the validity time frame as specified originally by the user.

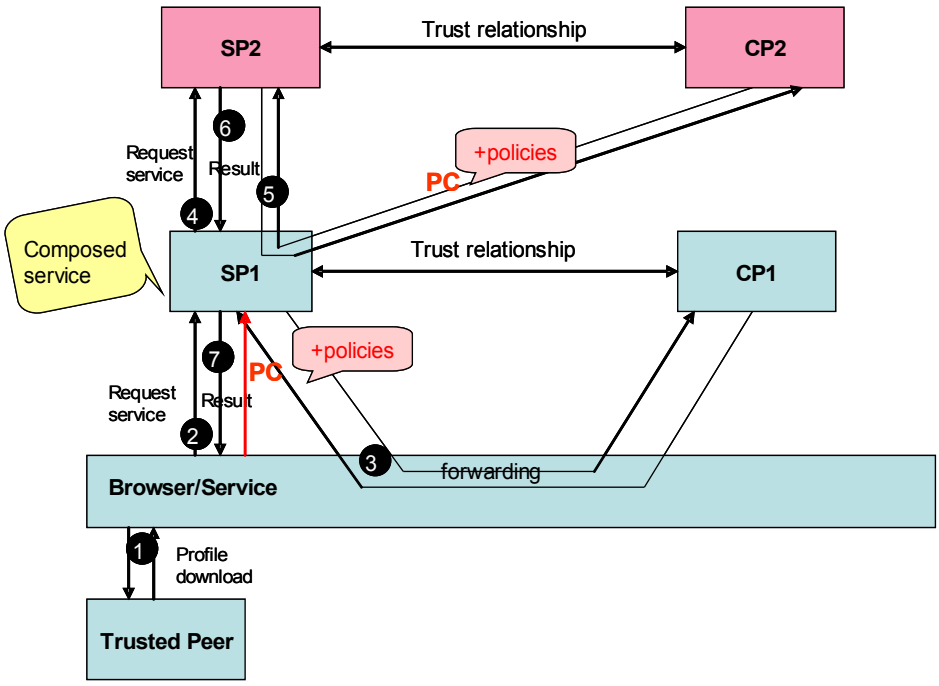


Figure 4: Service Composition using Proxy Certificates

Figure 4 shows the extended identity model for service composition. The steps behind the model are the following:

1. The user downloads the profile from a trusted peer that stores it.
2. The user requests a composite service from SP1.

3. The user is redirected to CP1 to logon (SSO use case). SP1 indicates that the requested service is an aggregation of services together with a list of the services to be used. The user identifies itself to the CP1 and then issues a PC to SP1 with policy that the proxy certificate will only be used for the scope of this service request and specified level of further aggregations.
4. SP1 requests a service to SP2.
5. SP2 redirects SP1 to CP2 for authentication. SP1 authenticates the user with CP2 using the proxy certificate (the PC obtained in step 3).
6. SP2 runs the service and provides the result to SP1.
7. SP1 completes the service execution and provides the result to the user.

The extended model scheme can be (recursively) applied in case SP2 needs to contact SP3 as next level aggregated service provider. Then SP2 takes the role of SP1 in the model.

4 Model Implementation

We produced an implementation that incorporates the core functionality of the model. We adopted a flexible approach to design, that would allow us to make significant changes to the communication scheme, or protocol, used to establish identity. We have endeavoured to make our design agnostic of specific technologies or underlying protocols, where possible, in order to make our implementation as broadly useful as possible.

4.1 Technologies used and design decisions

- Java is our choice of programming language. Java is widely used, has a rich set of class libraries, and is well suited to network applications.
- SAML v2.0 is the standard we use to communicate identity information between the entities in our DE. The second version of the protocol is more flexible and powerful than the first one. The SAML Single Sign-On (SSO) profile, using a combination of the HTTP GET and POST redirect bindings proved a good fit for our model.
- OpenSAML v2.0 beta (Java) is the SAML implementation we are currently using. Since SAML v2.0 is a new standard, implementations are naturally at an immature state. However, a full OpenSAML v2.0 release is due to be released in the near future.
- Although the model and implementation are agnostic of network protocols, we focussed on directly supporting thin-client (i.e. browser based) user agents and web (HTTP) based SPs and CPs. This scenario is highly restrictive in terms of the communication schemes we can use, and is therefore an excellent demonstration of the flexibility of our model and implementation.
- No user profile implementation. A pure SAML based implementation of the model (see below) obviates the strict requirement for a user profile. Therefore we do not give an implementation here. (A cache of SAML assertions might still prove useful, potentially used in tandem with certain credential provider specific tokens such as cookies for HTTP.)

4.2 Refining the model

The identity model is flexible and general enough to allow for many potential implementations. It is non-prescriptive in terms of the authentication mechanisms supported and the format of the security tokens that are passed between entities in the system. We chose to adopt a 'pure SAML' approach in our implementation (seeFigure 5). Using this approach, all identity information in the DE after the point of authentication is passed in the form of a SAML assertion. We use SAML assertions, passed between trusted entities, in place of arbitrary security tokens to “assert” identity, rather than passing the credentials themselves.

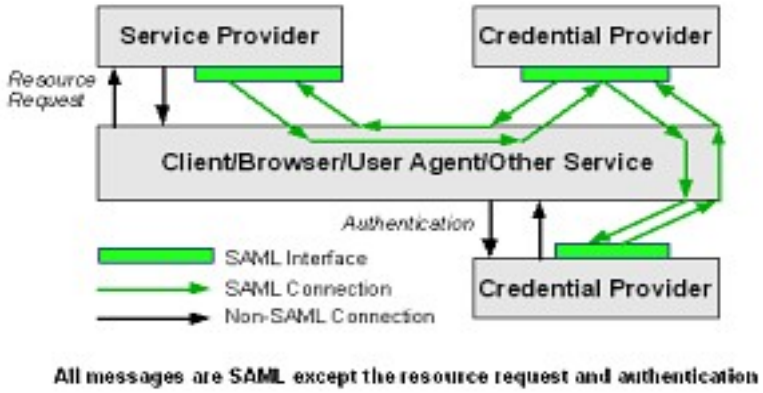


Figure 5: Implementation using a 'pure SAML' approach

This gives us the benefits of dealing with a single, simple “token”, rather than pandering to particular SPs and CPs by supporting the arbitrary tokens they might ordinarily use. The cost of adopting this approach is that a “SAML interface” must be exposed by SPs and CPs in order to interpret these SAML assertions and take the appropriate actions. We consider the benefits of a pure SAML approach, in terms of simplicity and extensibility, to be worth this cost. The cost itself can be reduced by providing powerful, generic interfaces for use with SPs and CPs that can be adapted easily to support a wide range of applications.

It is also worth mentioning that it may be of dubious benefit to support the generation and manipulation of a number of different types of security tokens, since there would be a continuous requirement to support new types of tokens, and no strong incentive to converge on a standard.

A significant real world use-case in identity is the scenario where a 'dumb' client such as a browser is the user agent whose identity an SP is attempting to establish. Such clients do not accept incoming requests (only responses to requests) and generally speaking do not have the intelligence to be anything more than passive entities in the protocol flow used to establish identity. Our model must be flexible enough to support such clients. If dumb clients are supported, we can guarantee that all clients that match or exceed a browser in sophistication will also be supported.

We use HTTP redirects in all instances where manual input from the user is not required. These redirects are illustrated by arrows that pass through, but do not terminate at, the user agent, as illustrated in Figure 7 below.

4.3 Actors and Connections

In our implementation, we model DE entities as Actors that participate in the protocol flow that establishes identity. We refer to a message transfer between two Actors as a Connection. We can describe any communication scheme used to establish identity in terms of these two elements. In terms of the identity model, there are three main Actors: the user agent (UA), the service provider (SP) and one or more identity or credential providers (CPs). Examples of Connections used to pass messages between Actors might be, DirectHTTPConnection or GETRedirectHTTPConnection. Actors and Connections can be extended arbitrarily.

In a real DE, Actors may co-exist on the same node (P2P), or machine. That is to say that an entity in a DE could behave as a UA, an SP and/or a CP at the same time, or at different times and/or in different contexts.

4.4 Building Operations from Profiles

We define an Operation as some protocol flow that establishes an identity related function on a network. Examples include Web Single Sign-On, Single Sign-Off, or an attribute request by an SP to a CP concerning a UA. Our identity model is concerned mainly with conducting SSO in a DE, but we use the more general concepts of Operation and Profile to make our model and implementation as flexible and extensible as possible.

Profiles are analogous to, and inspired by, SAML profiles. SAML profiles specify a skeleton protocol flow for identity operations. In our model, Profiles specify portions of the overall protocol

flow specified by an Operation. Therefore an Operation is built from a number of Profiles. Profiles are themselves build from Connections. And example of an Operation is given in Figure 5 above. Some of the Profiles in this Operation specify all their Connections, such as the SAML profile (since this Figure is derived from the SAML v2.0 profiles document), while other Profiles, such as the Authentication Profile have not been expanded.

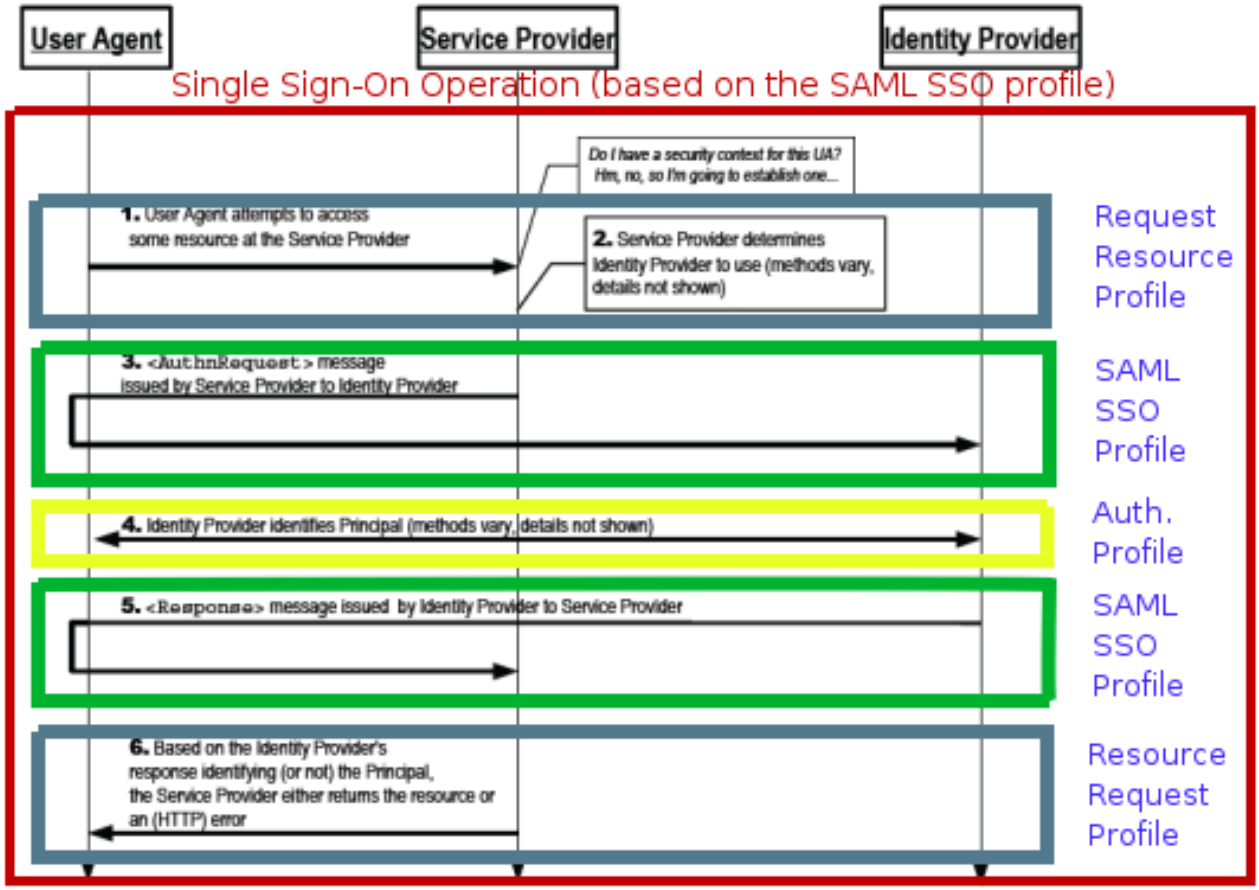


Figure 6: Single Sign-On Operation based on the SAML SSO profile

The flow of a protocol specified by an Operation cannot be pre-determined ahead of time, because its next step will often be contingent on events that occur during the protocol execution itself. For example, a SSO Operation will be conducted differently if a user enters a correct password to when a user enters an incorrect password. In the former case, the Operation may attempt to return a 'sign-on successful' type of result to the requester, while in the latter the Operation may attempt to divert the user to a 'try again' password screen. CPs have to make decisions as to whether they can assert a user's identity or not; here again, the Operation will progress differently depending on whether the

answer is a yes or a no. Because we must allow for dynamically determined contingencies in the protocol flow, Operations cannot be constructed from a simple series of Profiles (or Profiles from Connections) prior to protocol execution. They must be constructed instead from a contingent interconnection of Profiles, which are in turn constructed from a contingent interconnection of Connections.

This approach, although somewhat involved, allows us to represent an Operation as something that is reasonably easy to understand and design on the high level (since it lends itself to modelling), while encapsulating the resulting complexity at a lower level. It also gives us the maximum flexibility possible, since it does not tie us down to any particular protocol flow.

The most important Operation required by our identity model implementation is the Single Sign-On scenario. See Figure 6 above for a diagrammatic view of this Operation. In this implementation of the identity model, OpenID backs the UA's CP authentication, and trust is established dynamically between the UA's CP and the SP's CP by passing a token (potentially a SAML artefact) across a 'trusted path' of CPs.

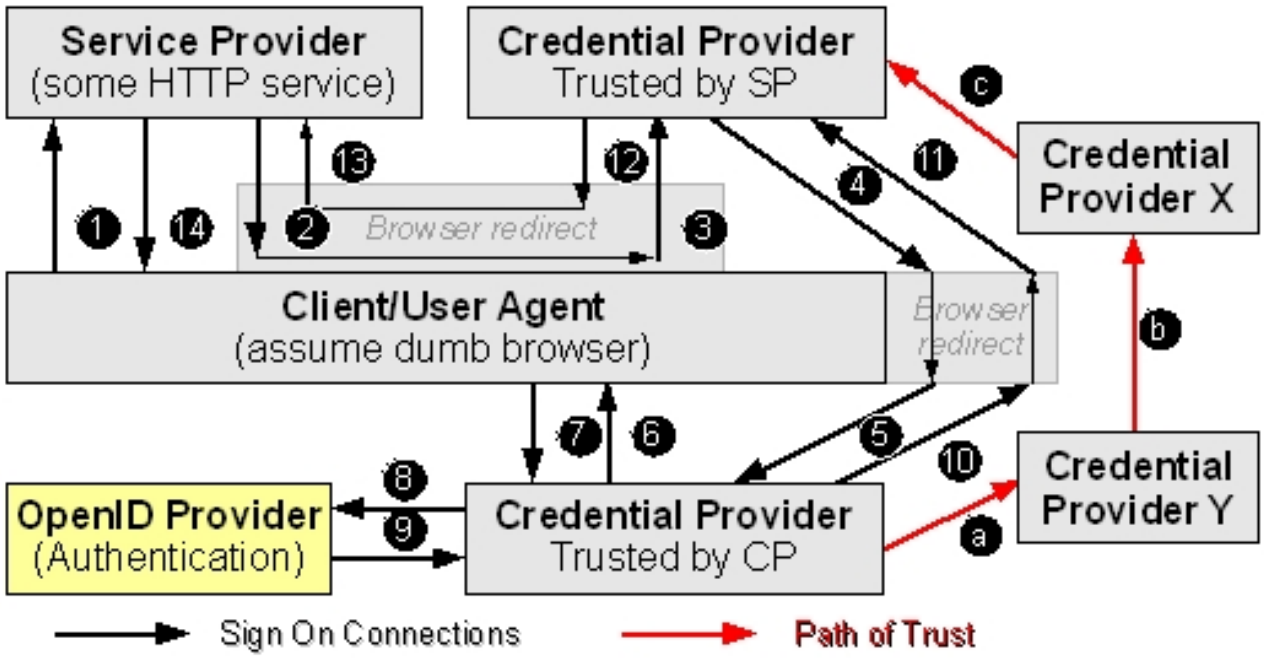


Figure 7: Single Sign-On Operation (with simplified view OpenID authentication)

4.5 Actors and State

Operations define the communication scheme or protocol flow. In order to make practical use of Operations we must express them from the point of view of the participating Actors.

Each Actor supporting a given Operation constructs a state machine representing the States (in the protocol flow) the Actor may occupy. Actors, with the exception of Actor initiating the Operation, begin in a 'waiting state', and progress from one state to another depending on dynamic factors determined during protocol execution.

The Operation, and hence the protocol flow, will come to an end when Actors reach pre-determined States.

The Single Sign-On Operation begins when the SP requests its CP to assert the UA's identity and ends when the SP receives a response from the CP. A broader view of the same Operation begins when a resource is requested that triggers an identity request, and ends when access to the resource is granted or refused.

4.6 Open Source Project - *IdentityFlow*

Software supporting the identity model implementation, as well as an SSO Operation and a demonstration, are being developed as part of an open source project, *IdentityFlow*[23]. Open source development is consistent with the spirit of the OPAALS project and helps facilitate the development of identity model implementations by making the source available to a wider community.

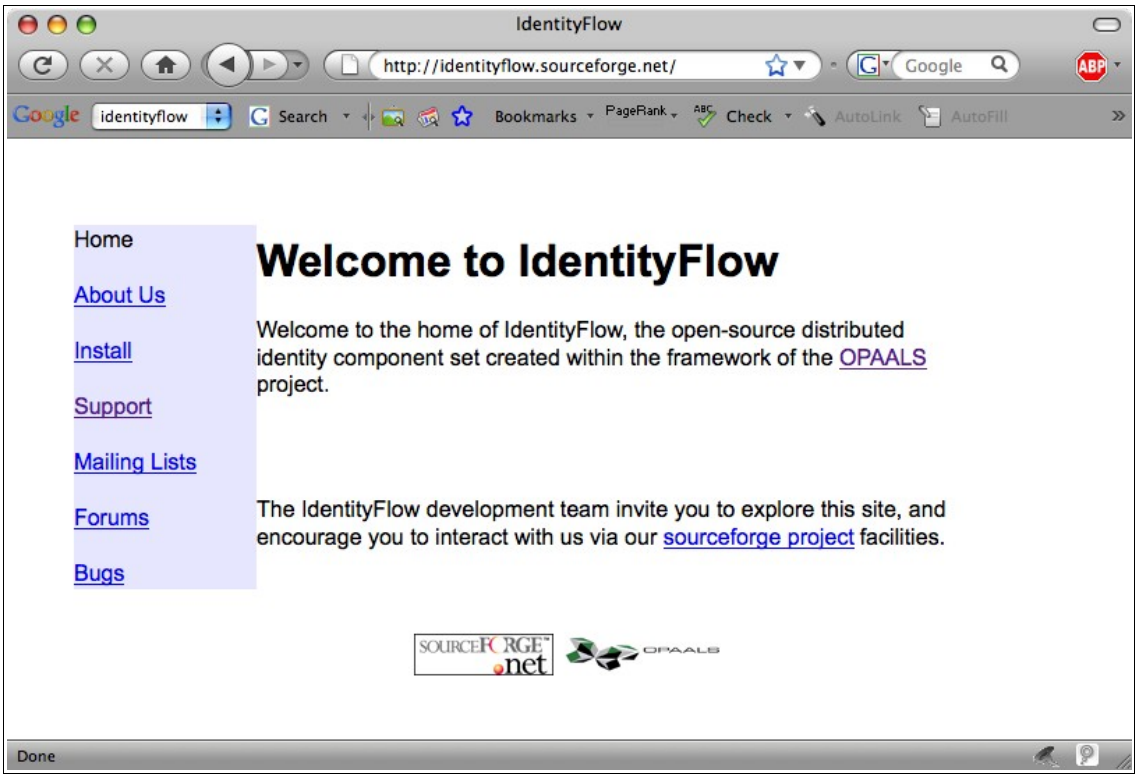
The software is written in Java with Servlet/JSP extensions as an example web interface to accept Actor Connections (SOAP/web services could be used, or an alternative mechanism).

A sourceforge.net project called *Identityflow*[2,3] has been started to host the project. The project contains a home page outlining a description of the project and useful links to project resources, a CVS repository containing code that is under development, a Maven 2 repository containing custom dependencies required to build the code, and immediately deployable software releases containing the identity model software implementation and demonstrations. The software currently consists of

[2] http://identityflow.sourceforge.net/

[3] http://sourceforge.net/projects/identityflow/

five sub-projects: Identity Model SAML, libraries for integrating the OpenSAML libraries with the identity model; Operation Builder, libraries for building Operations; Operation Request Handler, libraries for intercepting and handling incoming connections, including Servlet/JSP integration; Single Sign-On Operation, an implementation of an SSO Operation based on SAML profiles and HTTP GET/POST bindings; and Actors using SSO Example, which is a demo illustrating the use of the provided SSO Operation to identify a user agent to a service provider.



The build system used to build all of the software provided is maven[4]. With maven installed it is trivial to check out the latest code from the sourceforge.net CVS, build each sub-project and produce a Java web archive (war) file containing the demo application. The war file can be deployed immediately on a servlet container implementing the Servlet 2.5/JSP 2.1 specifications, such as Tomcat 6.0.

The demo, or sample application, provided, simulated a typical user-agent – service provider interaction, where the identity of the user-agent must be established by the service provider. The service provider initiates an SSO Operation in order to accomplish this. Each Actor is represented by a simple JSP page that redirects incoming Connections to an Interceptor. The Interceptor

[4]http://maven.apache.org/

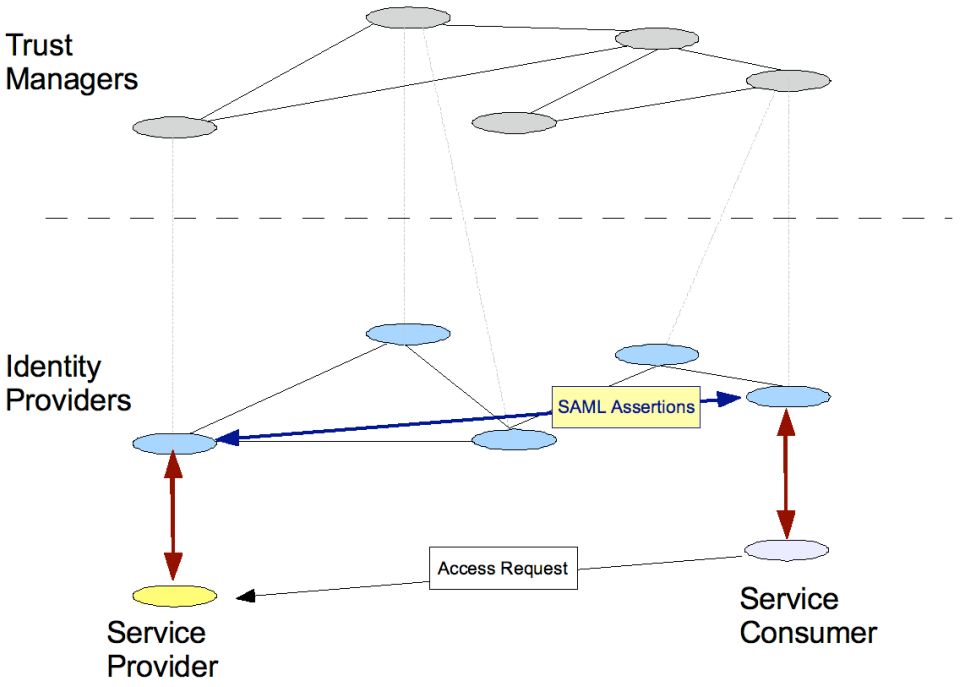
determines which Operation the Connection pertains to and forwards it to its OperationHandler. The OperationHandler then determines, based on the current state of the Actor and the incoming Connection, what it should do to continue protocol execution, and afterwards what outgoing Connections to other Actors it should initiate.

In the demo, the service provider's IdP cannot establish the identity of the user-agent so it refers to the user-agent's IdP. SAML assertions are then passed back to the service provider as directed by the SSO Operation, which establish the user-agent's identity to the service provider. OpenID is used as the backend identity provider for the user-agent's IdP. This demo, and the SSO Operation, will become more sophisticated over time as they exhibit a greater range of functionality.

4.7 Integrating Identity and Trust Models

Here we describe how the trust overlay described in deliverable D4.3 integrates with our identity model in ascertaining the trustworthiness of identity assertions. In Figure 9 a Service Consumer requests access to a service provided by Service Provider. Both of the entities have an associated IdP. The identity assertions work in both directions, in that both entities wish to ensure that the other is indeed who they say they are. In order to do so, the associated IdPs exchange SAML assertions concerning the identity of their respective entities.

The IdP will need to determine how trustworthy this assertion is and requests from the trust overlay a trust value for the other IdP. This trust value has evolved over time via the process described in Deliverable D4.3. With this trust value the IdP can derive the degree of certainty that the other entity is who they say they are. This degree of certainty is passed to the respective Service Provider/ Consumer and depending on whether this certainty is above or below a pre-determined threshold, the service access may proceed. For example, SP1 may have determined for access to a ServiceX, there must be a confidence of above 90% in the identity of the requesting entity (i.e. 0.9 probability that SC1 is who it says it is). Similarly SC1 might have set a threshold of 60% confidence that SP1 is who it says it is before it is willing to consume ServiceX. Such thresholds allow for a flexible degree of service access, for example in the case of an expensive financial transaction, the SP would most certainly need to be 100% certain that the consumer was who they said they were and vice versa, whereas in the case of access to information from a public forum there might not be a need for such a high threshold.


Trust
Managers
Identity
Providers
SAML Assertions
Access Request
Service
Provider
Service
Consumer

5 Concluding Remarks and Future Directions

In this deliverable we have presented a new identity management model for Digital Ecosystems. The model bridges main identity standards by using SAML as a unified message-level protocol for querying and obtaining authentication attributes. By using SAML, we are able to automate the process of identifying entities in a distributed environment. We suggested the use of user profiles to keep an abstract view of user's identity information such as certificates, username/passwords, public/private keys etc. To scale to service composition, we adopted the use of proxy certificates with three main policy settings limiting service scope, level of aggregation and validity. We have presented the extension of the model to address the problem of identity management and control for service compositions. The extended model provides the end-user with the ability to control the use of its identity information in case of service aggregations.

We have also presented a refinement of this model with a view to implementing a set of identity building blocks which allow ecosystem members to define their own identity protocol which satisfies their own requirements. The implementation of this work is being maintained in the open source project *IdentityFlow* where a simple example single sign on application is also provided. How this identity management approach is integrated with our overlay trust model (as described in D4.3) is also provided.

The trusts ratings and the accumulation of recognition within the network mirrors social processes of building trust. Rather than trying to define and centrally classify the identity of every user, this approach is based on the trustworthiness of identity assertions. This is much closer to a social model of trust than a technologically enabled one (in which large amounts of data can be classified, stored and searched). It also captures the dynamic relationship of trust in social networks where a trust breakdown leads to a sudden or gradual disassociation from a network. In terms of approach, this model of enabling trust relationships is much closer to a systems theory approach than it is a classical, scientific approach whose primary concern is with classification. From this point of view, focus rests on the dynamic 'sets of relations' formed rather than on categorically defining every object involved in a network of interactions. These research issues are pursued further in Deliverable 12.3.

[1] X.509. The directory: Public-key and attribute certificate frameworks, 2005. ITU-T Recommendation X.509:2005 | ISO/IEC 9594-8:2005.

[2] SPKI. SPKI certificate theory, 1999. IETF RFC 2693, www.ietf.org/rfc/rfc2693.txt.

[3] YADIS. Identity and Accountability Framework for Web 2.0. http://yadis.org/wiki/Main_Page

[4] OpenID Specifications, http://openid.net/developers/specs/

[5] Lightweight Identity, http://lid.netmesh.org/wiki/Main_Page

[6] Extensible Resource Identifier (XRI) Resolution Version 2.0, http://docs.oasis-open.org/xri/xri-resolution/2.0/specs/cd03/xri-resolution-V2.0-cd-03.html

[7] The XDI RDF Model version 10, http://www.oasis-open.org/committees/download.php/28080/xdi-rdf-model-v10.pdf

[8] iNames, http://www.inames.net/

[9] Windows CardSpace, http://msdn.microsoft.com/en-gb/netframework/aa663320.aspx

[10] Higgins Trust Framework Project, http://swik.net/higgins

[11] SXIP 2.0 protocol specification, http://sxip.net/index.php?title=Specs#SXIP_2.0_Protocol_Specification

[12] Digital Identity eXchange Protocol, http://dixs.org/index.php/Draft-merrells-dix-01.txt

[13] SAML. Security Assertion Markup Language (SAML), 2005. www.oasis-open.org/committees/security.

[14] XACML. eXtensible Access Control Markup Language (XACML), 2005. www.oasis-open.org/committees/xacml.

[15] Security Assertion Markup Language (SAML) Version 2.0 set of specifications, OASIS Standard, 15 March 2005. http://www.oasis-open.org/specs/index.php#samlv2.0.

[16] D-FF. Liberty Identity Federation Framework (ID-FF), 2007. www.projectliberty.org/resources/specifications.php.

[17] ID-WSF. Liberty Identity Web Services Framework (ID-WSF), 2007. www.projectliberty.org/resources/specifications.php.

[18] WS-Trust. Web Services Trust Language (WS-Trust), 2005. http://www-106.ibm.com/developerworks/library/specification/ws-trust.

[19] WS-Federation. Web Services Federation Language (WS-Federation), 2006. http://www-106.ibm.com/developerworks/webservices/library/wsfed.

[20] H. Koshutanski, M. Ion, and L. Telesca, "A distributed identity management model for digital ecosystems," in P*roceedings of International Conference on Emerging Security Information, Systems and Technologies* (SECURWARE'07). Valencia, Spain: IEEE press, October 2007.

[21] WS-Policy. Web Services Policy Framework (WS-Policy), 2004. http://www-106.ibm.com/developerworks/library/specification/wspolfram.

[22] Steven Tuecke, Von Welch, Doug Engert, Laura Perlman, and Mary Thompson. RFC3820: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, 2004. www.ietf.org/rfc/rfc3820.txt.

[23] IdentityFlow SourceForge Project, http://identityflow.sourceforge.net