

22.05.08

Deliverable DJ5.3.2: Architecture for Unified SSO



Deliverable DJ5.3.2

Contractual Date:	28/02/08
Actual Date:	22/05/08
Contract Number:	511082
Instrument type:	Integrated Infrastructure Initiative (I3)
Activity:	JRA5
Work Item:	WI3 (uSSO)
Nature of Deliverable:	R (Report)
Dissemination Level	PU (Public)
Lead Partner	University of Murcia
Document Code	GN2-08-080v2

Authors: Óscar Cánovas (UMU), Manuel Sánchez (UMU), Gabriel López (UMU), Ruth del Campo (USTUTT), Sascha Neinert (USTUTT), Jürgen Rauschenbach (DFN), Ian Thomson (DANTE).

Abstract

This deliverable describes the results of the DAME (Deploying Authorisation Mechanisms for Federated Services in the eduroam Architecture) project, which has two main goals; to integrate authorisation mechanisms for network properties into eduroam, and to realise unified Single Sign On (uSSO). This document describes example scenarios that demonstrate the project's concepts. It also explains the technological background, relevant Grid technologies, and the proposed architecture for uSSO. The results of deployment in a testbed are described, including setup details as well as the test results that demonstrate the successful validation of the proposed architecture.

Table of Contents

0	Executive Summary	v
1	Introduction	1
2	Scenarios and Use Cases	2
2.1	Basic Scenario: "Exchange Student"	2
2.2	Future Scenario: "Scientist"	3
3	Technical Background	4
3.1	eduroam	4
3.2	eduGAIN	5
3.3	NAS-SAML	6
3.4	User Attributes	7
3.5	Shibboleth	9
3.6	PAPI	10
3.7	XACML	11
3.8	Grid Computing	12
3.9	GridSphere Portal	13
3.10	VOMS	13
3.11	Globus Toolkit	13
3.11.1	Grid Authorisation Mechanisms	14
4	Architecture for Unified Single Sign On	16
4.1	Architectural Components	17
4.2	Network authentication and token delivery	18
4.3	Structure and Content of the eduToken	21
4.3.1	Using the eduToken for Multiple Consumers	22
4.4	Support for Multiple Federation Technologies	23
4.4.1	Shibboleth	23
4.4.2	PAPI	25
4.5	Data protection	27
5	Unified SSO Client Middleware	28
5.1	Supplicant	28

5.2	Token Manager	31
6	uSSO Profiles	33
6.1	Authorisation for Network Properties	33
6.2	Profile for Token-Based Web Authentication	35
6.2.1	Architectural Components	35
6.2.2	Message Flow	36
6.3	Unified SSO and Authorisation for Grid Computing	38
6.3.1	Architectural Components	38
6.3.2	Message Flow	39
7	Validation	42
7.1	Description of the testbed	42
7.2	Initial authentication and uSSO token delivery	45
7.3	Profile for network authorisation	48
7.4	Token-based Web Authentication	49
7.5	Unified SSO and Authorisation for Grid	51
8	Conclusions	54
9	References	55
10	Acronyms	57

Table of Figures

Figure 3.1: eduroam infrastructure	5
Figure 3.2. eduGAIN infrastructure	6
Figure 4.1: Proposed architecture	17
Figure 4.2: Network authentication	19
Figure 4.3: PEAP Authentication Method and Vendor-Specific TLV	21
Figure 4.4. Different SPs and IdPs in DAME	23
Figure 4.5. Conversion from the Shibboleth authentication assertion to the eduToken	25
Figure 4.6. Conversion from PAPI User's Assertions to the eduToken	27
Figure 5.1: Token packet format	31
Figure 6.1. Network authorisation	34
Figure 6.2. Token-Based Web Authentication	37
Figure 6.3: DAME Components of the Grid Scenario	39
Figure 6.4: Grid Authentication with eduToken	40
Figure 6.5: Grid Authorisation	41
Figure 7.1: uSSO testbed	43
Figure 7.2: Extended eduroam hierarchy	44
Figure 7.3: Time analysis. eduroam	46
Figure 7.4: Time analysis. Extended authentication	47
Figure 7.5: Extended authentication time components	47
Figure 7.6: a) Global time b) Network authz time components	49
Figure 7.7: Testbed unified SSO part	50

0 Executive Summary

This deliverable describes the results of the DAME project, which is a subproject of the Joint Research Activity 5 “Roaming and Authorisation” within GÉANT2. DAME (Deploying Authorisation Mechanisms for Federated Services in the eduroam Architecture) has two main goals:

1. To integrate authorisation mechanisms for network properties into eduroam.
2. To realise unified Single Sign On (uSSO).

The first goal addresses the need of institutions that (would) participate in the eduroam roaming confederation, but currently are unable to configure fine-grained, role-based network access policies according to their users’ needs and conforming to their local and national regulations. The second goal addresses the problem of providing SSO in heterogeneous environments as deployed currently in European institutions in a cross-layer approach, linking the roaming authentication to the web-AAI authentication.

This document begins by describing two example scenarios that demonstrate the DAME concepts; one basic scenario is described to illustrate essential features, followed by a future scenario that shows features that could be enabled later. Next the technological background is explained, most importantly eduroam as this is the foundation of DAME. eduGAIN and NAS-SAML are also described - how their components are linked to eduroam, providing support for policy-based authorisation, and for supporting an uSSO profile based on tokens. XACML is the language used to express the access policies. Relevant Grid technologies are portrayed, also.

The components and sequence diagrams detailing the proposed “Architecture for Unified Single Sign On” are shown next, beginning with those for authorisation for network properties and the delivery of the SSO token called eduToken. This is built on top of the existing eduroam infrastructure. The unified SSO middleware is presented afterwards, including new client software and a profile for token-based web authentication using eduGAIN. How to use this new architecture with an application-level Grid service is also presented.

Finally, a prototypical implementation of the DAME architecture is described. All these components have been deployed in a testbed that is federated to provide a realistic environment. Test cases are defined and the setups (as well as the test results) are documented, showing the successful validation of this architecture.

1 Introduction

The current eduroam infrastructure allows roaming users to gain wireless network access across all Europe, and beyond. Its basic principle is that authentication is done by the home institution and authorisation by the visited institution. Thereby it forms a federated Authentication and Authorisation Infrastructure (AAI). But the capabilities of RADIUS to perform this authorisation are rather limited, and eduroam Service Providers do not get much information about their users; they only see their name (sometimes not even that) and their home domain. This is much less than Service Providers for web-based AAIs (for example Shibboleth or PAPI) get to see. These can request user attributes as expressed in a schema as eduPerson or SCHAC; for example the user's home organisation, the home organisation's type and the user's affiliation to that organisation. Using such information the Service Provider can perform authorisation based on groups and roles, rather than individuals. Certain Grid-based service infrastructures also make use of these fine-grained authorisation capabilities and define their own, sometimes application-specific, attributes. Such fine-grained attribute-based authorisation can be beneficial for both the users and the Service Providers. The users can get personalised access to extended services, which may or may not be offered to all "anonymous" users. The Service Providers can better protect their resources compared to letting every "anonymous" get access, but still can apply a much more scalable authorisation mechanism compared to individual access policies.

The idea of Single Sign On (SSO) is to get users to be authenticated only once, and then let them access all resources they are entitled to. Current SSO solutions are usually only applicable to a certain class of services, for example Shibboleth offers SSO for accessing web resources. But if one federation offers wireless network access (as eduroam does) and then would also offer access to web resources even across heterogeneous AAI middleware (as eduGAIN does), it would be possible to link both to require only one authentication of the user. This would be a "unified SSO" solution, which requires authentication for the first resource only (the network) and uses the authenticated state for accessing application-level services, web-based or also Grid services. Unified SSO improves the user experience as less password typing is needed, and could also possibly improve security as sensitive information needs to be transmitted less often. The uSSO architecture presented here is based on the requirements identified and defined in [USSOREQ].

How authorisation for network properties and unified SSO can be achieved using the eduroam and eduGAIN infrastructures is described in the following sections.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

2 Scenarios and Use Cases

2.1 Basic Scenario: “Exchange Student”

This scenario demonstrates the basic features that eduroam gains through the DAME project. It shows the advantages that DAME brings and that will be demonstrated in the testbed.

Isabel Gonzalez is a Master student from Spain participating in a Stuttgart-Murcia double diploma course. Both University of Murcia and University of Stuttgart are members of the European academic confederation.

Isabel arrived just a few days ago and goes to the computer lab’s helpdesk to register for an account. Unfortunately several dozen students are there waiting already. She decides to try again later and work on the documents she has stored locally on her laptop.

She takes a seat in a quiet corner of the cafeteria and opens her laptop. She is positively surprised when her wireless supplicant she also uses at home shows her a familiar SSID “eduroam”. She signs on to eduroam using her home-identity (isabel.gonzalez@um.es) and gets network access. First she checks for new mail using her mail-client, then she visits the university’s e-learning portal. Conveniently it offers a button “login with home account” that she selects. Without further interaction she is redirected to her personal site on the portal. It is in Spanish, but she wants to practice a foreign language and switches to the German version. Here she can look at her courses for the next week and also register for new ones. She has a question about a specific course and tries to contact the tutor, but unfortunately she cannot access the website with all the personal information as it is for internal use only; it tells her “restricted – for staff of University of Stuttgart only”. So she just writes an e-mail.

It is not apparent to Isabel that she would not be able to use any peer-to-peer client with her acquired network connection, but she would not have used such anyway.

In the course of the Bologna process it can be expected to have many more “Isabels” tomorrow than there are today, but it is unlikely that the money available to spend for helpdesk support and account administration will increase proportionally. However, with the federated identity management infrastructure that is illustrated here an increase in funds won’t be necessary. Also this infrastructure increases the ease-of-use and security as there is only one password to remember and enter, which is transmitted once. Users no longer need to repeatedly enter passwords for each and any resource (network, web resource, Grid resource, and so on) they

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

want to access. This benefit becomes more noticeable the longer and more secure passwords are used, and yet even more if other and more complex methods of authentication are employed. It also helps users to handle their credentials more consciously as they enter them only in one, known to be safe, location, instead of numerous ones of unknown trustworthiness. Authentication of the provider is possible, but is often not automated and a cumbersome task to the end-user.

2.2 Future Scenario: “Scientist”

This scenario shows more advanced features of DAME that could be made possible, including some that are yet to be designed. Thus it also shows a vision of features that might be built once the basic DAME infrastructure exists.

Karl Schmidt works as a scientist for the University of Stuttgart and is visiting a conference held at the University of Murcia. He wants to present a newly developed software client that allows him to access a virtual laboratory.

Before this demonstration he opens his laptop and sees a list of available networks. He requests information about the one called “eduroam” using his supplicant, and receives a list of properties digitally signed by an authority he knows and trusts; the network supports medium and high-level encryption and end-to-end security. Anonymous use is restricted to VPN, non-anonymous use does not require this. The network is provided free of cost for academic users. Up-to-date antivirus software is required as well as a dedicated client, and both can be obtained locally. He decides to give it a try. Before getting network access his antivirus client is given the latest signatures, and he is presented with a screen explaining how to install the preconfigured client.

After installation Karl starts the client, and his account at his home university serves to provide him the network access at the visited campus.

He starts his grid client, and sees that there is an option pre-selected to login using the same account he just chose for getting network access. As his university account is linked with his membership in the virtual organisation of his grid project, he is granted access.

Having tested that everything will be working fine, he has still some spare time until the conference begins. He starts his virtual community client to chat with a colleague. He could have used it for a telephone call without any problems, as the network connection given to employees of partner universities supports quality of service, but there are already several others in the room he does not want to disturb. The virtual community client shows that Karl’s colleague is in Zurich and also online. During the chat he is addressed virtually by someone not yet on his contact list. He looks up that person’s profile and sees a list of attributes, among the trusted ones is also the location - the very same room! He adds his new acquaintance to his contact list as he also works on similar areas.

This example shows some ideas how a unified and federated identity infrastructure could be experienced. Users would be enabled to make sensible decisions without requiring technical know-how, and use and manage their virtual identities consciously.

3 Technical Background

The uSSO architecture consists of several elements based on a wide range of technologies. Some of them are widely-accepted standards that are being deployed in several initiatives related to identity management and federations. Other technologies, which are related to mobility, interconnection of federations and authorisation, have been previously developed by TERENA, GN2 and University of Murcia. The technologies presented in this chapter were chosen because they are (partially) used for building and validating the uSSO architecture. This section provides some information about them.

3.1 eduroam

eduroam¹ (Education Roaming) [ER] is an inter-institutional roaming service based on the 802.1X architecture [8021X] and a hierarchical RADIUS-based infrastructure [RAD] defined by TERENA. This initiative allows users of participating institutions to access the Internet at other participants' institutions using their home institution's credentials, all with minimal administrative overhead. Depending on local policies at the visited institutions, eduroam participants may also have additional resources at their disposal.

The top level servers of the RADIUS hierarchy are operated by the SA5 Operational Team as part of the GÉANT2 project, and all the National Research and Education Networks (NRENs) belonging to the eduroam infrastructure are connected to these confederation RADIUS servers. Finally, each institution willing to join eduroam connects its own RADIUS server to the national server of its NREN.

¹ eduroam is a registered trademark of TERENA

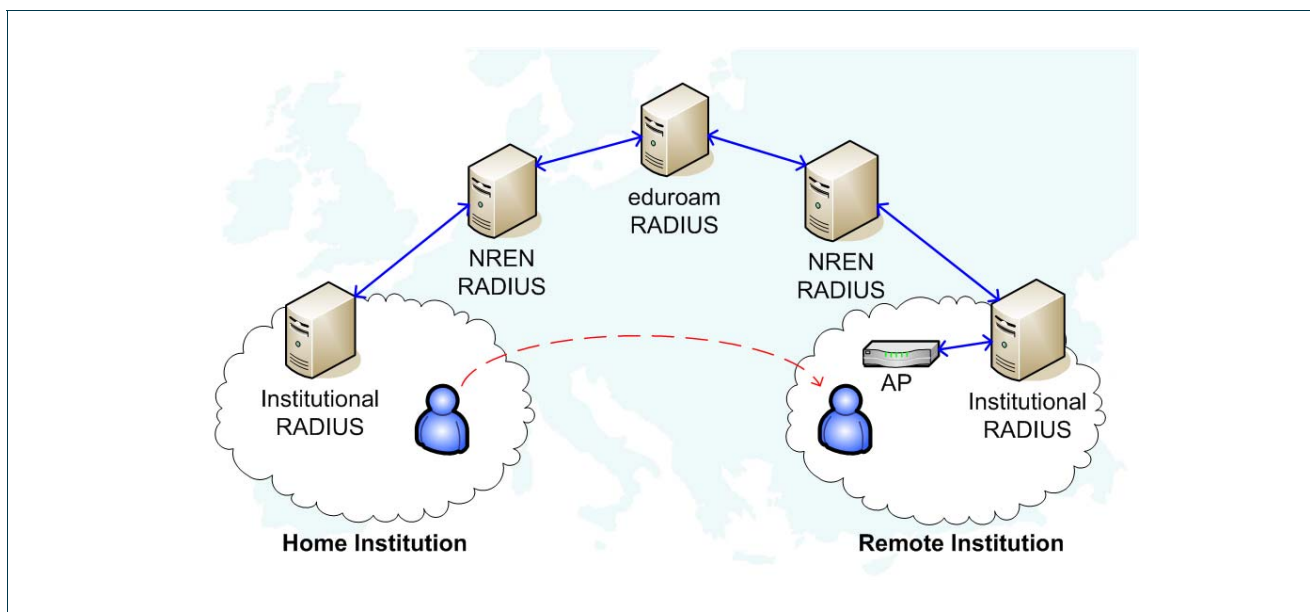


Figure 3.1: eduroam infrastructure

The scenario shown in Figure 3.1 depicts a user from Home Institution, who wants to get access to the wireless network in Remote Institution, both belonging to the eduroam federation. In this situation, access control is carried out following the 802.1X standard. The user associates with the wireless access point (AP), which contacts its local RADIUS server in order to authenticate the user. Once this server identifies that the user belongs to a different domain (based on the user identifier, for example) the authentication request is forwarded through the RADIUS hierarchy until the server in the user's home institution is reached. Then, the user is authenticated and the response is routed back to Remote Institution, where the AP enables the requested connection.

The 802.1X specification defines the use of EAP (Extensible Authentication Protocol) [EAP] for authentication purposes because it allows different authentication mechanisms. Therefore, institutions can use any of the different authentication methods depending on the required security level. For example, users can make use of login and password (EAP-TTLS), or digital certificates may be required both for users and servers (EAP-TLS).

Nowadays, more than 500 institutions in 29 countries in Europe, and some in the Asia-Pacific region, participate in the eduroam initiative.

3.2 eduGAIN

The main goal of eduGAIN [EG] is to build an interoperable authentication and authorisation infrastructure to interconnect existing federations. In this way, eduGAIN will provide means for finding the federation to which a roaming user belongs (MetaData Service MDS), for translating the messages between the internal protocols of the federation and eduGAIN and vice versa, and will guarantee trust among the participating institutions.

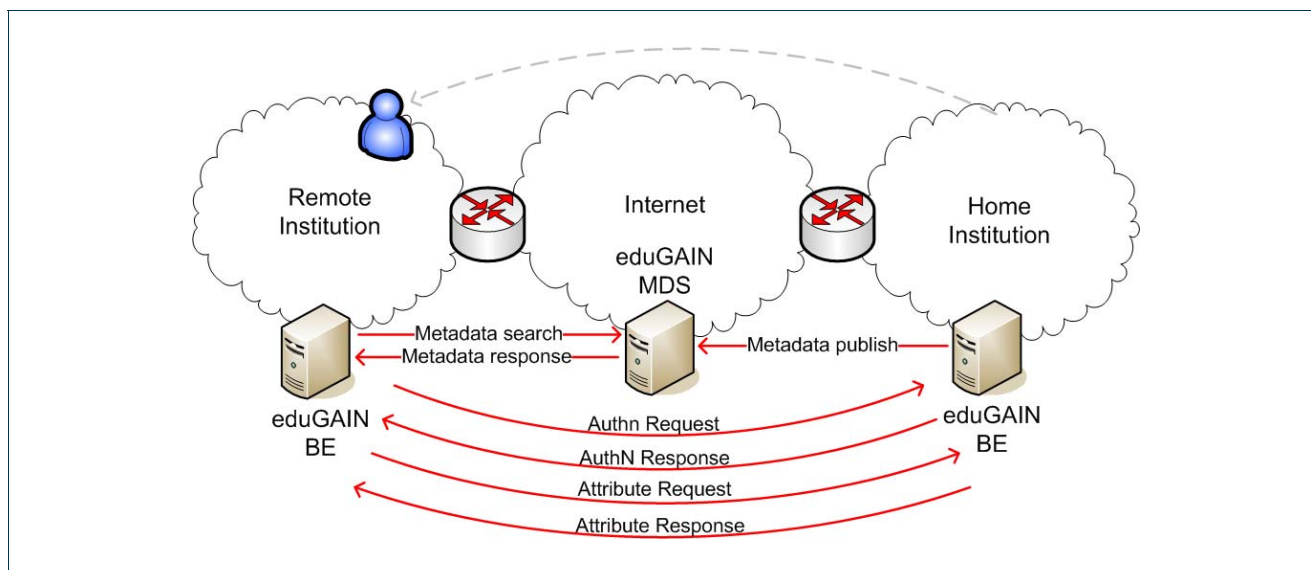


Figure 3.2. eduGAIN infrastructure

The main goal is achieved by defining a set of common services that include the MetaData Service (MDS), and a Bridging Element (BE), which is a confederation-aware element that is responsible for connecting the different federations to eduGAIN. In eduGAIN, metadata related to federations are published by means of the MDS. These metadata include information for locating the authentication and authorisation points of the federation. In this way, the home federation of a roaming user is located by the BE, which obtains the information published in the MDS. The appropriate authentication and authorisation requests are then translated and routed by the remote BE to the user's home institution. This scheme is also valid for communication between different institutions that belong to the same federation.

The way that the authentication and authorisation processes are carried out in eduGAIN is defined by different profiles according to the requirements from GÉANT2 activities. Currently, a profile compatible with Shibboleth (Web SSO), another one that does not require human intervention (Automated Client), as well as a Web Container (WE) and a User behind a Client (UbC) have been defined. The profiles AC, UbC and WE are all dedicated to perfSONAR requirements. Additionally, an eduGAIN filter was developed for AutoBAHN (IDC cooperation).

3.3 NAS-SAML

NAS-SAML [NAS] is a network access control approach based on authorisation attributes and a flexible, scalable and manageable authorisation system. The proposal is based on the SAML (Security Assertion Markup Language) [SAML] and the XACML (eXtensible Access Control Markup Language) [XACML] standards, which are used for expressing access-control policies based on attributes, authorisation statements and authorisation protocols. Authorisation is mainly based on the definition of access-control policies, including the sets of users pertaining to different subject domains that can be assigned to different roles in order to gain access to the network of a service provider, under specific circumstances. The starting point is a network scenario based on the 802.1X standard and the AAA (Authentication, Authorisation and Accounting)

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

architecture [AAA], where all the operations related to authentication, authorisation and accounting are centralised.

Briefly, the system operates as follows:

Every end user belongs to a home domain, where they were given a set of attributes stating the roles they play. When the end user requests a network connection in a particular domain by means of an 802.1X connection, the request is obtained by the AAA server. After authenticating the user by means of the corresponding authentication method based on EAP, the AAA server makes a query to obtain the attributes linked to the user from an authority responsible for managing them. Alternatively, the user can present their attributes instead of letting the AAA server recover them. Finally, the AAA server sends an authorisation query to a policy decision point, and that element provides an answer indicating whether the attributes satisfy the resource access policy. Furthermore, that policy can also establish the set of obligations derived from that decision, for example, some QoS properties or security options. This general scheme works both in single and inter-domain scenarios, and uses both push- and pull-based communications.

This solution constitutes an excellent starting point for providing Single Sign On (SSO) services to a wider set of applications than the network access control. In fact it can be used to establish a distributed architecture for maintaining and exchanging user profiles that can be used to provide security-enhanced application services. The key elements of that architecture will be the SAML assertions and protocols and the XACML policies [POLI] for expressing access-control restrictions. Moreover, since it is based on SAML, it can be easily integrated with other SAML-based authentication solutions, such as Shibboleth.

3.4 User Attributes

Attributes are most commonly of a descriptive nature, and associate a characteristic with an entity. Attributes associated with the person that initiates an action are called user attributes. Those that define attributes of resources are called resource attributes, while those that describe attributes of the environment are called environment attributes. This section focuses on user attributes and their relationship with the authorisation process.

User attributes provide information about users that can be used in addition to or instead of the user's unique identity to make authorisation decisions. Such attributes are used in combination with access policy to offer rights to specific resources.

User attributes will be selected based on the needs acquired in an eduroam, eduGAIN [AAICOOK], Shibboleth and DAME background. The selection of the user attributes will be realistic (the usual attributes in education should be the first to be covered) and taking into account privacy issues.

There already exists a starting point when dealing with user attributes in universities. There has been a lot of work done by several research groups aligned with the technologies (Shibboleth, eduGAIN) that the DAME project uses.

eduroam has not previously focused on user attributes. The messages exchanged for decisions about authentication between different eduroam parties state simply whether an access request has been accepted or denied. However, Shibboleth and eduGAIN do exchange user attributes messages. Shibboleth relies on the eduPerson [EDUPER] Schema, a schema for higher educational user attributes defined by Internet2 and EDUCAUSE initiative. EduGAIN, on the other hand, has focused its user attributes interest on the schema defined by the Schema Harmonisation Committee. SCHAC [SCHAC] schema that has been defined by the TERENA initiative with the aim of carrying out work in the area of attributes coordination. Thanks to the definition of the eduPerson and SCHAC, higher educational institutional directories are provided with a common list of attributes and definitions, which is valuable for inter-institutional data exchange.

Both eduPerson and SCHAC precisely define the names of the attributes, and in many (but not in all) cases also the syntax of the values to be signed, often with an additional definition of a set of allowed values. Always some descriptive text is given as a kind of meta-information, trying to describe the “semantics” of the attribute in natural language. A machine-readable form of these semantics is still the objective of current research. In a multi-national context with several languages, cultural backgrounds and an unknown but probably high number of application-specific requirements on attributes, this is hard to solve in general. Therefore a pragmatic approach was chosen in eduPerson and SCHAC; to start with a smaller set that is common, which can later-on be extended.

Additionally, eduPerson and SCHAC are not independent of each other. SCHAC, which was developed after eduPerson had been released, covers the eduPerson schema as a subset of its defined attributes. SCHAC covers specific defined attributes through to other general and already defined attribute schemas. This relationship benefits inter-institutional interoperability between SCHAC and eduPerson compliant institutions.

The selected attributes for DAME and for uSSO purposes is an initial list that may grow. The uSSO profiles presented in this document, and developed through the DAME project, focus on a roaming environment where user attributes are used to provide access control on the network, and also on the attributes exchanged in Web environments and Grid Computing.

There are at least two main reasons (non exclusive) why any attribute of this list (from either an eduGAIN or eduroam background) have been selected:

- The use of unique, distinguishing characteristics from users that can work as policy control. For example, the age of a user can be used to perform policy enforcement. Australian laws restrict Internet access to people over 18 years old [AUPOL]. Sending the user's date of birth enables the age-check required to access to the Internet without revealing any other attribute of the user.
- The possible use of service personalisation for the user. For example, the preferred language attribute can be used to show information to the user, like network statistics of his network access. All messages could be sent in the selected desired language of the user to make it easily understood.

The following list shows the attributes selected either from application services or from roaming environments:

- eduPersonPrincipalName
- eduPersonScopedAffiliation
- preferredLanguage
- eduPersonEntitlement

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

- eduPersonTargetedId
- schacsn1
- schacsn2
- schacDateOfBirth
- schacCountryOfCitizenship
- schacHomeOrganisation
- schacPersonalPosition
- schacExpiryDate
- schacUserPrivateAttribute
- schacUserStatus

3.5 Shibboleth

Shibboleth [SHIBB] defines an access control system for the Web by means of a SAML-based authorisation infrastructure. The main goal is to exchange information about a user, who is surfing the net through its Web browser, to determine if they are allowed access to a target resource. This information about the user, which can be represented as attributes, is defined in their home domain.

In this way, Shibboleth enables the creation of identity federations in such a way that different domains (identity and service providers) exchange information about the user's identity in a reliable way. Because authorisation decisions are based on the user's attributes, it is not necessary to disclose the user's identity.

The architecture is composed of three basic entities:

- Service providers (SP), which offer resources.
- Identity providers (IdP), which maintain the user's credentials
- Where Are You From (WAYF), which lists all Identity Providers belonging to the federation

When a user tries to access to a resource protected by a SP, the SP asks the suitable IdP for information about the user's authentication status. In this way, the user only needs to authenticate once, providing a Single Sign On (SSO) environment. Optionally, the SP can request the user's attributes defined in the IdP. This process is transparent to the user, by means of Web redirections.

The Identity Provider is composed of the following elements:

- SSO Service: This is the entity that authenticates the user. The user is redirected here from the SP, and once authenticated is redirected to the Inter-Site Transfer Service to ask for an authentication credential.
- Inter-Site Transfer Service: This acts as a gateway between the SSO Service and the Authentication Authority.
- Authentication Authority: This entity issues SAML authentication statements that will be gathered by the SP.
- Attribute Authority: Once the SP knows that the user is authenticated, it can request attributes from this entity.

The Service Provider in Shibboleth is composed of the following elements:

- Assertion Consumer Service: This is the service responsible for protecting the resource. It sends the authentication request to the SSO Service and interprets the authentication statements received.
- Attribute Requester: Once the user is authenticated, this entity is responsible for requesting the user's attributes from the Attribute Authority of the IdP.

Shibboleth uses a service called WAYF (Where Are You From) to locate the user's IdP with or without user interaction. It is independent of both the SP and the IdP. This element is essentially a proxy for the authentication request passed from the SP to the SSO Service.

Therefore, the authentication process is as follows:

- The user attempts to access a resource protected by a SP through its web browser.
- Then, the SP redirects the user to the SSO Service of their IdP, using the WAYF if the corresponding IdP is unknown beforehand.
- The SSO Service authenticates the user and redirects them to the Inter-Site transfer Service to request the Authentication Authority for a SAML authentication assertion from the Authentication Authority.
- This assertion is then sent to the Assertion Consumer Service of the SP. This service checks the authentication assertion and, if it is necessary, requests its Attribute Requester Service to acquire the user's attributes.
- Finally, based on the authentication assertion and the user's attributes, the decision is made whether to allow access or not.

3.6 PAPI

PAPI is a system for providing access control to restricted information resources across the Internet. It keeps authentication local to the organisation the user belongs to, while leaving the information providers full control over the resources they offer.

The authentication mechanisms are designed to be as flexible as possible, allowing each organisation to use its own authentication schema (thereby keeping user privacy), while offering target sites enough data to perform authorisation and collect usage statistics. Moreover, access control mechanisms are transparent to the user and fully compatible with the most commonly employed Web browsers and any operating system.

The system consists of two independent elements:

- The Authentication Server (AS).
- The Point of Access (PoA).

This structure makes the final system much more flexible and able to be integrated into different environments. There is no need for a one-to-one mapping between AS's and PoAs; a given PoA can deal with requests from any number of AS's and direct them to any number of web servers.

The purpose of the AS is to provide users with a single authentication point and enable them to retrieve (in a completely transparent manner) the temporary keys that will give them access to the services they are authorised for. A PAPI2 AS may also include an Attribute Authority (AA) server associated with it, able to provide requesting PoAs with those relevant attributes related to the user that is trying to access a given resource.

The PoA manages authorisation and actual access control to a set of web locations for a given organisation. The owner of the target site has the responsibility of managing this point of access.

With PAPI, both authentication and authorisation are handled locally. Authentication occurs at the user's organisation, possibly accessing sensitive data that must not be disclosed. Once authenticated, the user is able to go the entry points of the PoAs. It is important to note that the AS is not sending any user-provided data to any PoA. It prepares an assertion (as required by the PoA) about the user and signs it using its private key. The only constraint that a PoA imposes on the assertion about a user sent by an AS is that the identifier must be unique (at least during the lifetime of the tokens the PoA is going to provide). The PoA receives this information, signed by the AS, and decides whether to grant access to the user or not.

3.7 XACML

XACML (eXtensible Access Control Markup Language) [XACML], the OASIS proposal for a standard access control language, was defined to represent access control policies in a standard way. XACML is XML-based and includes two different specifications:

- An access control policy language: This defines the set of subjects that can perform particular actions on a subset of resources.
- A representation format to encode access control requests and responses: This is a way to express queries about whether a particular access should be allowed, and the related answers.

The main element of all XACML policies is a *Policy* or *PolicySet* element. A *PolicySet* is a container that can hold other *Policies* or *PolicySets*, as well as references to other policies (*PolicyIDReference*).

A *Policy* represents a single access control policy, expressed by a set of *Rules*. A *Policy* or *PolicySet* may contain multiple policies or *Rules*, each of which may result in different access control decisions. XACML needs some way of reconciling the decisions each makes, and this is done through a collection of *Combining Algorithms*. An example of those is the *Permit Overrides Algorithm*, which says that if at least an evaluation returns *Permit*, then the final result is also *Permit*. A *Policy* or *PolicySet* element may also specify a set of *Obligation Attributes*; that is, a set of actions to be performed if the request has been approved.

XACML provides another feature called *Target*. This is a set of simplified conditions for the Subject, Resource and Action that must be met for a *PolicySet*, *Policy* or *Rule* to apply to a given request. If all the conditions of a *Target* are met, then its associated *PolicySet*, *Policy*, or *Rule* is applied to the request. Once a *Policy* is found, the included rules are evaluated. The main element of a rule is the *Condition*. If the evaluation of the *Condition* results true, then the *Rule's Effect* (*Permit* or *Deny*) is returned.

The main object that XACML deals with is attributes. Attributes are named values of known types. Specifically, attributes are characteristics of the Subject, Resource, Action or Environment in which the access request is made.

As described in 6.1 "Authorisation for Network Properties", XACML is the language that has been used to specify the authorisation policies for network access control.

3.8 Grid Computing

Grid computing is a service oriented distributed computing model that provides the ability to perform higher throughput computing by taking advantage of many networked computers (considered as Grid resources that execute jobs) in order to model a virtual computer architecture that is able to distribute process execution across a parallel infrastructure. The above definition of the Grid computing was brought together by Ian Foster, Carl Kesselman and Steve Tuecke, who also led the effort to initiate the standardisation of Grid computing infrastructure.

In general, a Grid Computing infrastructure component must address several potentially complicated areas, from middleware to Grid application services. In summary the areas considered as the pillars of the Grid computing architecture are:

- Resource management: This is the process of job assignments and monitoring of Grid resources.
- Information services: Making informed decisions on resource assignments based on the current utilisation, which is maintained and provided by the Monitoring and Discovery Service.
- Data management: Making decisions for accessing shared storage based on the size of the data repositories, security requirements and the underlying technologies utilised for storage and data access.
- Security: Accessing Grid resources with different security policies that need to be complied with.

In addition to this, Grid enabled application services are located above the middleware infrastructure. Application services must use the required interface to manipulate and manage the Grid middleware infrastructure.

The emergence of service provider-oriented architectures and the open and extremely powerful utility value of XML-based interoperable messages has resulted in the merging of traditional Grid computing with web services architecture. The Open Grid Services Architecture [OGSA] specifications developed by the Global Grid Forum and Web Service Resource Framework specifications (now an OASIS [WSRF] standard) are the result of this merge. The goal of OGSA is to standardise all the services found in a Grid application (job management services, resource management services, security services, and so on) by specifying a set of standard interfaces for these services. The requisite of statefull web services of OGSA has led to the development of WSRF [WSRF] specifications.

3.9 GridSphere Portal

GridSphere [GS] is a multi-national project developing an open source framework for the development and deployment of web portals. On top of this framework, Grid applications can be developed and then provided as part of Grid portal. These Grid applications are portlets compliant to the Portlet API as documented in JSR (Java Specification Request) 168. GridSphere implements this standard and includes a set of core portlets and portlet services that provide the basic infrastructure required for developing and administering Web portals. A number of existing portlets are available; for example, for allowing end-users to make use of Grid technologies, administering virtual organisations, and managing Grid resources.

Within a GridSphere portal any number of grid portlets can run and be executed on behalf of the users without requiring grid clients to be installed on the users' devices. In this way users have a standard way of accessing grid computing services. Without a portal, instead of a browser a typical grid client would need to be used (often command line based, requiring users to have advanced knowledge of commands). With a grid portal, the access is web-based, and users can access their grid services in a much easier way.

The GridSphere framework has been "shibbolised" by the Australian MAMS (Meta Access Management System) project. This supports several use cases, such as inviting Shibboleth users so they can afterwards sign on to a specific GridSphere portal using their home institution's credentials. This modified GridSphere can easily be employed to perform federated authentication of users in the DAME infrastructure.

3.10 VOMS

The Virtual Organization Membership Service (VOMS) manages information describing the user's relationship with his Virtual Organization. The attributes can be membership in groups, roles and capabilities, for example. It uses X.509 Attribute Certificates as a means to express the authorization information. Using VOMS an attribute-based authorization to the Globus Toolkit job management services is enabled. The VOMS is not used in the Grid scenario for validating the uSSO architecture (see chapter 7.5) because the Grid use case is with ViroLab, which does not use VOMS. If unified SSO should be added to Grid architecture using VOMS, possibly some interfaces of the components would need to be adapted, but otherwise this uSSO architecture is generic and applicable to that as well.

3.11 Globus Toolkit

Focusing on Grid implementations, the Globus Toolkit [GT] is the leading implementation software. It implements both OGSA and WSRF specifications. It is the most complete product that provides software, APIs and tools for the development of Grid infrastructure and application services. Due to the consolidated position that the Globus Toolkit possesses in Grid Computing environments (even more in scientific environments), the DAME project focuses on this toolkit to provide Grid authorisation based on user attributes.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

Globus Toolkit version 4 contains implementations of WSRF (Web Service Resource Framework), a family of OASIS-published specifications for web services. WSRF provides a set of operations for possibly stateful web services; web service clients communicate with resource services that allow data to be stored and retrieved. The part of GT4 that enables the WSRF protocol is WS-Core, which is a WSRF container for Grid Services developed with this protocol.

Globus Toolkit enables WSRF protocol and one part of the Globus software called WS-Core contains a WSRF container for Grid Services developed with the WSRF protocol.

One of the pillars of the Globus Toolkit is the Grid Security Infrastructure (GSI), which provides many facilities to help manage the security requirements of the Grid environment. Among the main functionalities included at the GSI are:

- **Authentication:** GSI uses X.509 end entity certificates (EEC) to identify persistent entities, such as users and services. X.509 EECs provide each entity with a unique identifier and a method to assert that identifier to another party through the use of an asymmetric key pair bound to the identifier by the certificate. Since Globus Toolkit 4, GSI also supports authentication through plain username and passwords combined with WS-Security. Authentication with X.509 Credentials can be accomplished either via TLS, in the case of transport-level security, or via signature as specified by WS-Security, in the case of message-level security.
- **Authorisation:** This is described in the following section. GSI offers two kinds of authorisation based services; grid-mapfile and SAML Authorisation statements. The grid-map file, which has been used widely, provides access control based on a list of acceptable user identifiers. The SAML AuthorisationDecision assertions provide external authorisation decision services for access control requests to GT4-based services.

3.11.1 Grid Authorisation Mechanisms

From the very beginning of Grid computing, user authorisation has been a priority. The authorisation process (that is, the right of a user to perform an action in Grid) has always been an important requirement.

In Globus Toolkit, the authorisation framework consists of an engine that evaluates a chain of Policy Decision Points to determine if the Grid client making the invocation can access the resource/service. The chain can also be configured with Policy Information Points (PIPs), with the functionality to collect attributes from the invocation context, which can be used to glean information that would be useful in the decision making process. The following table shows some of the PDPs that can be configured within the Globus Toolkit environment.

PDP Type	Description
None	No authorisation performed

Self	Only clients that present the same identity as the identity in the current subject associated with the service are allowed to access the service
Identity	The client identity must match the value of a system property set to the expected identity.
Host	The host should match the expected system property.
Gridmap	Grid map file authorisation is performed, i.e. a mapping must exist for the client identity in the configured grid map file.

Although many possibilities of PDP types are possible in Globus Toolkit, they all deal with static identity information. For instance, many Grid Authorisation PDPs have been configured with the well known grid-mapfile. A grid-mapfile contains the information of the user's DN (from the X.509 certificate) bound to a local user of the system or to a virtual group of the system.

The static information needed to be stored in each system makes the grid-mapfile authorisation mechanism a non-scalable solution. Also, all systems require the real identity of the user, not allowing anonymous access, resulting in a very intrusive method.

Since Grid developers and architects were aware of the disadvantages of a grid-mapfiles, new trends in authorisation have appeared. In Globus Toolkit v3 (GT3) there is the ability to customise an external authorisation framework to Grid services. This ability conforms to the OGSA specifications about authorisation services.

Plugging in a custom authorisation module allows for the replacement of the usual GT authorisation mechanisms with an arbitrary method. It is possible to enable any service with its own authorisation logic by developing custom interceptors and enabling them in a security descriptor. It could be used, for example, to integrate the Globus Toolkit with a third-party authorisation. In such a way, interface classes for the Policy Decision Point and Policy Information Point must be implemented in order to be linked to the current desired authorisation service.

Additionally, GT4 currently enables the use a SAML Callout, the SAML callout authorisation. PDP contacts a SAML authorisation service by means of OGSA compliant SAML request/response authorisation messages. Because of the standard message formats, different authorisation services can be plugged in to allow different authorisation clients. The work of the Grid SAML callout has been carried out by the OGSA Authorisation Working Group. The aim of this Working Group is to define the specifications needed to allow for basic interoperability of authorisation components in the OGSA framework.

4 Architecture for Unified Single Sign On

Taking into account current eduroam infrastructure, it would be interesting to create a seamless link between the network-layer authentication mechanism and any additional authentication step that will be needed when users try to gain access to application-level resources. This would involve extension of the network access mechanism in order to deliver additional information (some kind of security credentials) that might be used at the service level in order to avoid further user re-authentication.

In order to define an architecture for uSSO, a necessary first step is the delivery of a security token during network access that will be used later to avoid unnecessary re-authentication. Once that token has been transmitted to the user, it will be necessary to define how an application-level service would be able to validate that information using a federation-level service. Section 6 defines some profiles based on eduGAIN to access to protected resources.

The architecture presented in this paper follows the guidelines defined in [USSOREQ]. These are the main requirements that have been considered:

- The user has the appropriate credentials in his home institution (HI).
- HI belongs to a federation where its issued credentials are valid.
- When the user tries to access a resource (R) in another institution (RI), the decision about the access is taken by an Authorisation Service at the RI.
- The authentication and authorisation information about users is exchanged between institutions through a mechanism provided by the confederation.
- There is a confederation-aware component called "Local Federation Adaptor" (LFA), which decides if an authentication request can be handled locally or must be sent to another institution. The functionality related to this LFA is commonly performed by the eduGAIN BE.

When using eduroam, once the user is associated with a wireless access point or is connected to a wired switch in the remote institution, they are asked for authentication information. When provided, those credentials are forwarded to the user's home institution to be authenticated. To enable the SSO, an authentication service is responsible for providing some kind of statement to the user that can be used later to achieve the SSO.

Later, the user might want access to a protected resource in this institution or in another one belonging to the same federation. Then the user's credentials are sent to the resource in order to be validated to gain access. Depending on the type of resource, additional steps for obtaining user attributes would be needed.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

4.1 Architectural Components

The eduroam network is mainly composed of a RADIUS hierarchy of authentication servers deployed in each institution. The integration of uSSO mechanisms during the network access requires new functional elements. However, eduroam is an already deployed network with hundreds of institutions using it. Therefore it is necessary to introduce changes gradually, maintaining backward compatibility and allowing institutions to introduce the new functionality step-by-step. For a longer period there will be institutions that not yet having deployed the new components, and their users want to use eduroam as now, and also these institutions want to offer eduroam as now.

To address these issues, the proposed architecture extends the authentication phase. That is, once the user is authenticated through the eduroam network, the home institution provides the authentication token for SSO purposes.

Following the eduroam nomenclature, the Home Institution (HI) is where the mobile user belongs to. It is responsible for performing the authentication process and, for authorisation purposes, when other institutions request information about users it must release only the appropriate attributes following a specific policy. The Remote Institution (RI) is where the roaming user is trying to access to the network. It has to determine the properties related to the network connection of the visiting users, according to specific attributes. Figure 4.1 shows this architecture and the required elements:

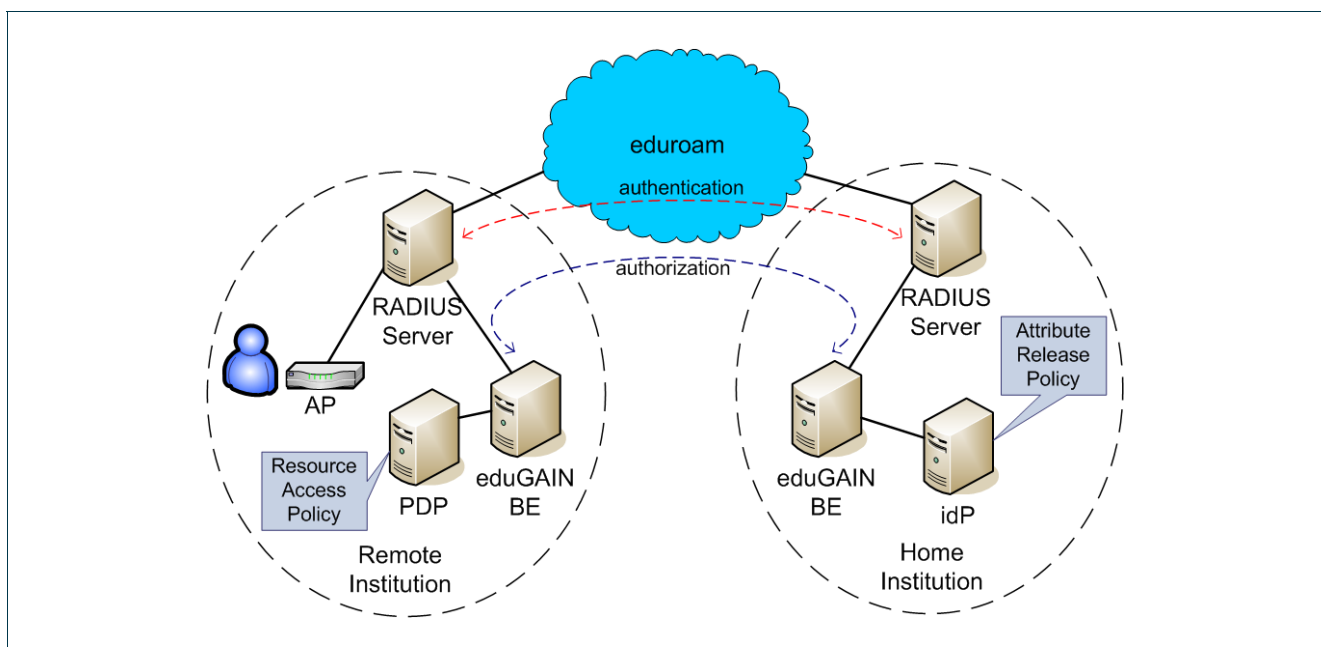


Figure 4.1: Proposed architecture

- **RADIUS server in RI:** In eduroam, this server receives authentication requests from the access point (802.1X). When the user belongs to the visited organisation the request is processed locally. Otherwise,

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

the request will be forwarded to the appropriate home RADIUS server. As part of the proposal presented in this document, the RADIUS server will be extended to support the authorisation phase in order to request authorisation decisions based on the user attributes, once the user is authenticated. In order to provide a common framework, the authorisation phase will be performed through the eduGAIN infrastructure.

- **RADIUS server in HI:** In eduroam, this server receives authentication requests from remote domains, including user authentication credentials that are validated locally. For SSO purposes, it has to be extended to query the Identity Provider (IdP) for an authentication statement, once it has performed its own authentication of the user. In this case, the IdP acts as a central point for both authentication and authorisation purposes. Together with the authentication statement, the RADIUS server will receive a handle (token), which it forwards to RI. If required, this handle will be used afterwards by the remote RADIUS server to request the user attributes. Finally, IdPs are institution dependent, so in order to provide a common interface between the RADIUS server and the IdP, the communication is carried out by means of the eduGAIN infrastructure (BE).
- **BE in HI.** Following the eduGAIN architecture, BEs are common points to request authentications and authorisations statements. During the authentication phase, as previously described, we propose to use this element in HI to provide a common interface between the RADIUS server and the local IdP, in order to make IdP specific details transparent to the server. Taking advantage of the already defined BE functionality, this element will be extended in order to generate and manage the authentication handle described above. During the authorisation phase, it will receive attribute requests from RI. Those requests include the handle as proof of identity, which will be used by the IdP to select the user attributes to be revealed.
- **BE in RI.** This component acts as the bridge between both institutions during the authorisation phase. The remote RADIUS server, once the user is authenticated, will ask the local BE about the user attributes, defined in this home domain, in order to be able to take the right decision. Once the BE has obtained those attributes, it will query the PDP entity for the obligations or properties to be applied to the network connection, and will forward those properties to the RADIUS server. The BE has to be extended to support the communication with the RADIUS server and to manage the authentication handle. In order to allow an easy integration and the minimum impact over eduroam, the selected communication protocol between RADIUS and BEs is LDAP.
- **Policy Decision Point (PDP).** This new module is responsible for taking the authorisation decisions about users based on a *Resource Access Policy*. Basically, this policy defines which user attributes are required to enforce specific properties or obligations in some resources. The PDP receives the requests from its local BE, making this process transparent to the RADIUS server.
- **Identity Provider (IdP).** This module is responsible for providing information about the users belonging to the institution, and it is institution dependent. During the authentication phase, this module is in charge of generating authentication statements and handles, and later on, it will be able to disclose the user attributes by means of the authentication handles.

4.2 Network authentication and token delivery

To provide one unified Single Sign On (uSSO), authentication for services is bootstrapped from the network authentication. This is done by delivering a token to the client during the network authentication phase. The delivery of the token imposes several demands on the architecture described previously:

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

- First, the token must be generated. Its generation must originate at the entity that later can reply to attribute requests based on this proof, that is, the identity provider (IdP). Therefore:
 - A trusted link between the RADIUS server in the home institution and the IdP must be established in order to allow the former to request the token to the latter. This trusted link can be a static one.
 - Finally, the token must be sent to the user through some secure channel to avoid security problems.

Figure 4.2 shows the network authentication phase and token delivery proposed by DAME. This phase starts when the user requests access to the eduroam network in a remote institution. In the usual case, the user presents his authentication credentials, which are composed of his email address and password. The remote RADIUS server, by means of the email domain identifier, recognises the user as a foreign user and, through the eduroam network, redirects them to their home institution.

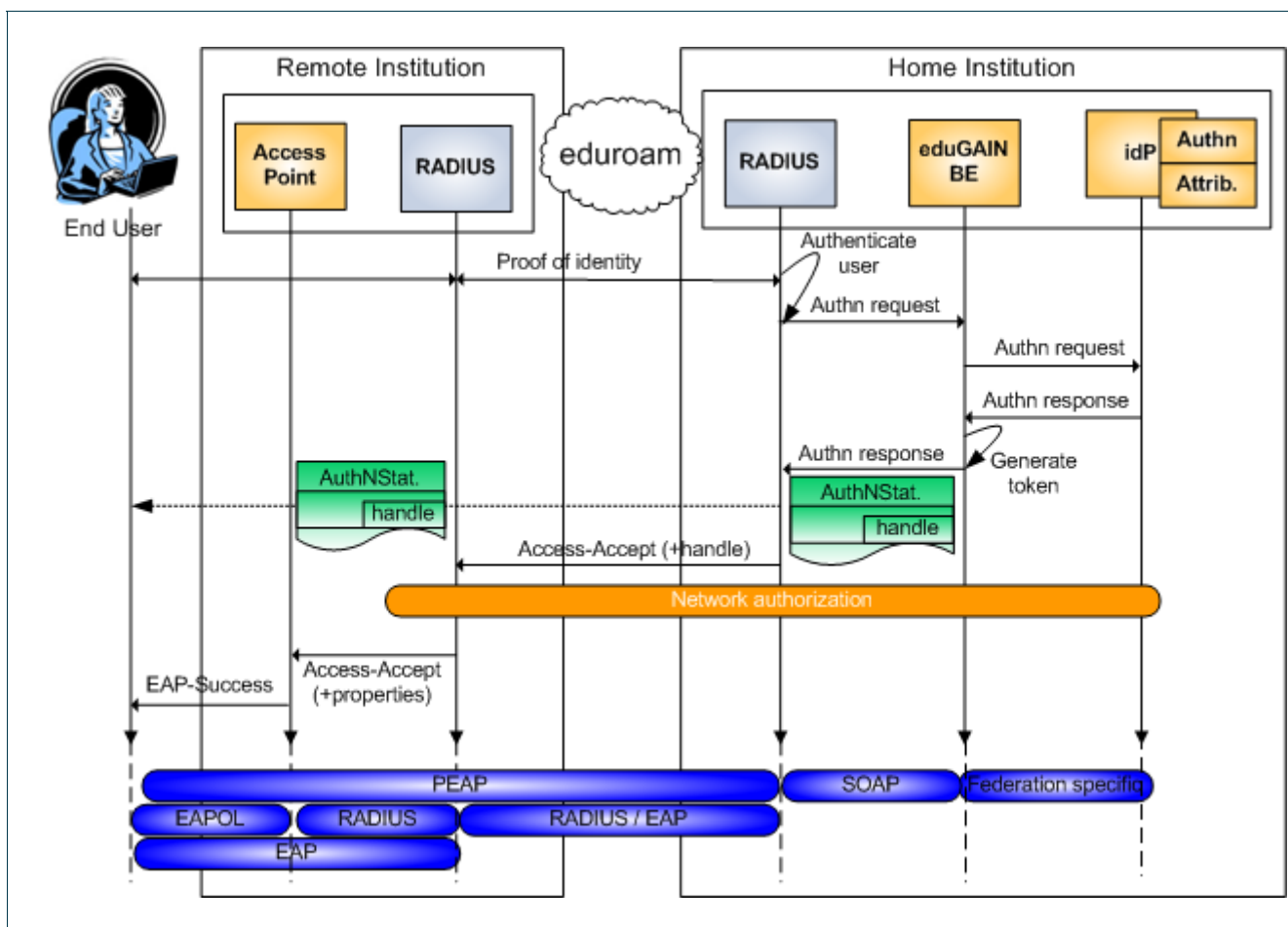


Figure 4.2: Network authentication

The home RADIUS server authenticates the user exactly as in standard eduroam. After that it invokes, by means of a SAML *AuthenticationQuery* message, the IdP in order to get an authentication statement. This query includes the user's subject and the authentication method used by the server. This step is necessary in order to link the authentication statement with a further attribute query, and this information is centralised in the

IdP. As previously described, the communication between the RADIUS server and IdP is done by means of the BE, using HTTP/SOAP as communication protocol.

The IdP generates a SAML *AuthenticationStatement* sentence and sends it back to the BE. The communication between these modules will depend on the IdP implementation. For example, if Shibboleth or NAS-SAML are used then this communication is made through HTTP/SOAP.

Once the home BE receives the authentication statement, it will generate an authentication statement including a handle. The authentication statement will be used as a token and the handle will be used during the network authorisation phase. The remote BE forwards the response (including the new handle) to the RADIUS server, which will send a RADIUS *Access-Accept* message to the remote RADIUS server. The handle, usually the user's subject, can be forwarded to the remote institution in the *Access-Accept* message.

As detailed in Figure 4.3, a TLS tunnel is established by the authentication method. That tunnel is part of the PEAPv2 [PEAP] authentication method. The reason for using PEAPv2 to authenticate the user is the need for a protected channel to deliver the SSO statement. Figure 4.3 shows the sequence of messages needed to authenticate the user using this method. It has the special feature that, after an initial handshake, it creates a protected tunnel between the peer and the authenticator. Then, this tunnel is used to exchange the authentication information in a secure way.

The uSSO mechanism can take advantage of this tunnel and use it to deliver the statement to the user in a secure way. Specifically, elements are transmitted through the tunnel by means of *Type Length Value* (TLV) objects. There is also a special, vendor-specific TLV that can be used to carry non-standard elements. In this way, the authenticator can add the security data inside a vendor-specific TLV to the last message sent to the peer before closing the tunnel.

It is important to note that the architecture presented here is not limited to PEAP. Other EAP types providing an end-to-end secured tunnel as EAP-TTLS can also be used. From the architectural view, both methods are equal. Added support for EAP-TTLS might, however, imply further modifications on the RADIUS module requesting the token from the IdP on the remote site, and also on the supplicant receiving the token. So, for the implementation of the first prototype, the PEAP method was chosen.

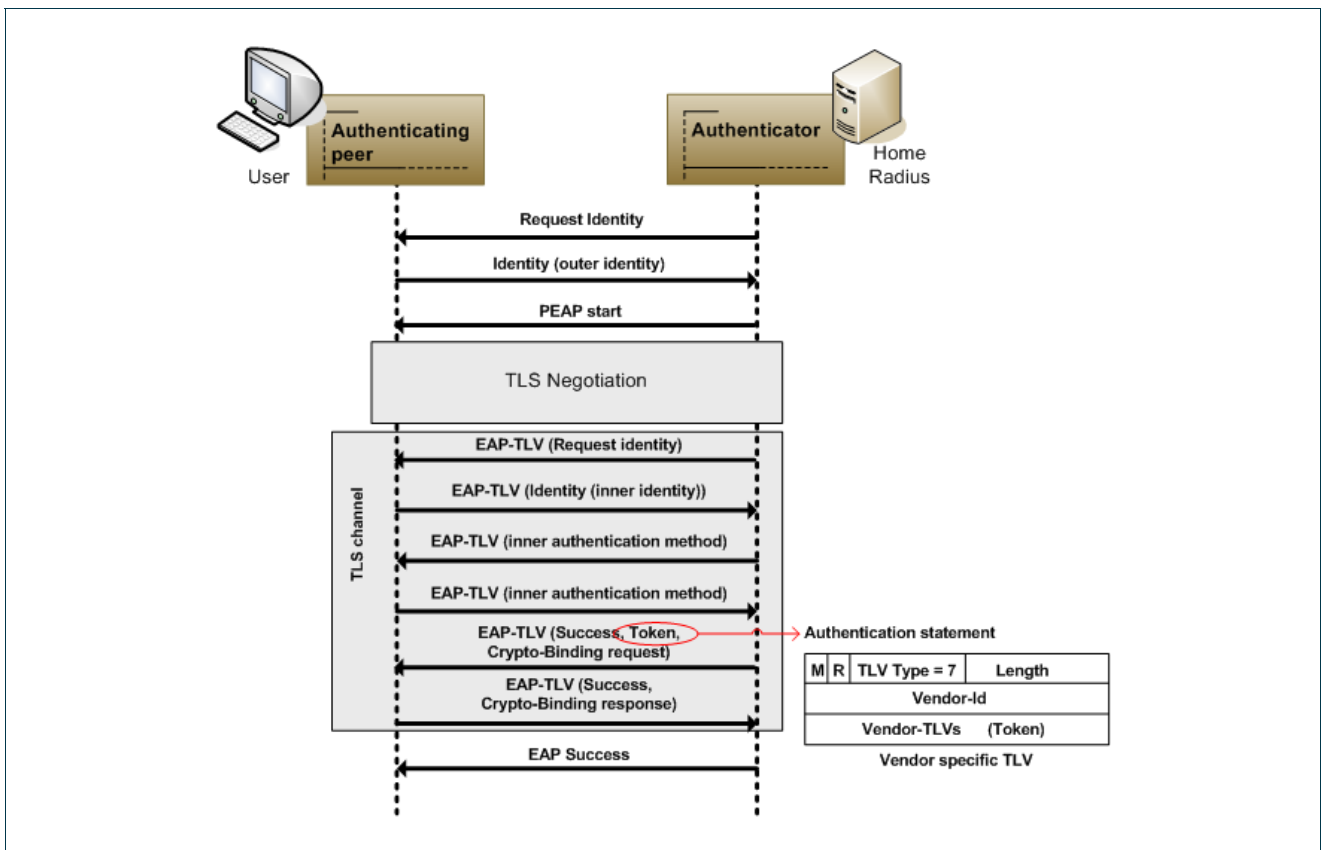


Figure 4.3: PEAP Authentication Method and Vendor-Specific TLV

4.3 Structure and Content of the eduToken

The token, which is the piece of data the user receives to enable uSSO, represents that this user has been authenticated by some trusted entity. Therefore it needs to contain information about this entity, and the date and time when this information is valid. To facilitate subsequent attribute queries, it must also contain the user's identity. Therefore, due to this architecture being based on eduGAIN to exchange the user's attributes, it is reasonable to use an eduGAIN statement as the SSO token. The eduGAIN *AuthNResponse* attribute fulfils these requirements and so was chosen and renamed *eduToken*. Furthermore, the protocol for requesting attributes in eduGAIN is based on the use of this statement to request the user's attributes. In this way, once the user owns this statement, the protocols already defined in eduGAIN can be used to request the user's attributes.

This statement includes the handle that identifies the user, and can contain an attributes list. Specifically, the eduGAIN *AuthNResponse* is composed of the following fields:

- ResponseId: sequence number.
- ProducerId: The component identifier for the producer of the response.
- ConsumerId(1): The component identifier for the consumer of the response.
- NotBefore: The date after which this response will be valid.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

- NotAfter: The date after which this response will no longer be valid.
- InResponseTo(2): reference to the authentication request that originated this response.
- Result: result of the process, for example *Accepted*, *RedirectUserTo* or *ConnectTo*.
- SubjectHandle(*): user's identifier.
- AttributeValueList*(*) (3): list of attributes to return after the authentication.
- Interfaces(*): list of URLs which can be of interest if the result is *ConnectTo* or *RedirectUserTo*.
- AdditionalData(*): this field can hold other information needed.

(*) These parameters are optional.

(1) The token is not to be used for one specific consumer only.

(2) There was no eduGAIN request before this response: not used.

(3) The current version of the token does not contain attributes.

In the binding of eduGAIN to SAML, the *AuthNResponse* is encoded as a SAML *Response* containing two assertions. The first one is mandatory and contains a SAML *AuthnStatement*, and the second one is optional and contains a SAML *AttributeStatement*. In this way, the mapping of the previously defined fields with these SAML elements is as follows:

eduGAIN	SAML
ResponseId	Response/ResponseId
ProducerId	Out of direct SAML mapping. Corresponds to the SubjectAlternateName extension in the certificate used for the XML signature of both assertions: Response/ds:KeyInfo/ds:X509Data/ds:X509Certificate
ConsumerId	Response/Assertion/Conditions/AudienceRestrictionCondition/Audience
NotBefore	Response/Assertion/Conditions@NotBefore
NotAfter	Response/Assertion/Conditions@NotOnOrAfter
InResponseTo	Response/InResponseTo
Result	Response/Status/StatusCode/Value
SubjectHandle	Response/Assertion/AuthnStatement/Subject/NameIdentifier
AttributeValueList	Response/Assertion/Assertion/AttributeStatement
Interfaces	Response/Status/StatusMessage
AdditionalData	Response/Status/StatusDetail

4.3.1 Using the eduToken for Multiple Consumers

The eduToken serves as proof that the holder of it has been authenticated by a trusted entity within the (con)federation, which is the issuer of the eduToken. As in other federated SSO solutions the user does not need to be authenticated a second time when going to a second SP, so the principle of using this kind of information at multiple Consumers (SPs) is not new. The difference with the DAME infrastructure is the bigger scope where the token is valid – it comes from the European eduroam confederation and is to be used within the European eduGAIN confederation. So a stolen eduToken can cause as much harm as stolen eduGAIN credentials in the form of plain username and password.

In a simple realisation the ConsumerId (in SAML: AudienceRestrictionCondition/Audience) would simply not be used. If it is to be expected that more than one (con)federation is operational that could accept the eduToken, the ConsumerId should probably be set to some URI defining either the (con)federation, or all Consumers within that (con)federation.

It is of course possible for remote BEs to reject such multi-consumer tokens and not offer this unified SSO profile to their customers if the ordinary authentication via username and password would be considered more secure for some reason (the uSSO profile would then be an optional eduGAIN profile, not mandatory).

4.4 Support for Multiple Federation Technologies

The proposed uSSO architecture must support the use of different federation technologies, for example Shibboleth or PAPI. As Figure 4.4 shows, a common core based on eduGAIN is necessary, along with some way to connect to different IdPs. As defined in eduGAIN, the element responsible for connecting local federations to the confederation is the BE, therefore one BE is necessary for each different federation idP. This BE is responsible for translating the authentication and authorisation messages between the internal format of the federation and eduGAIN.

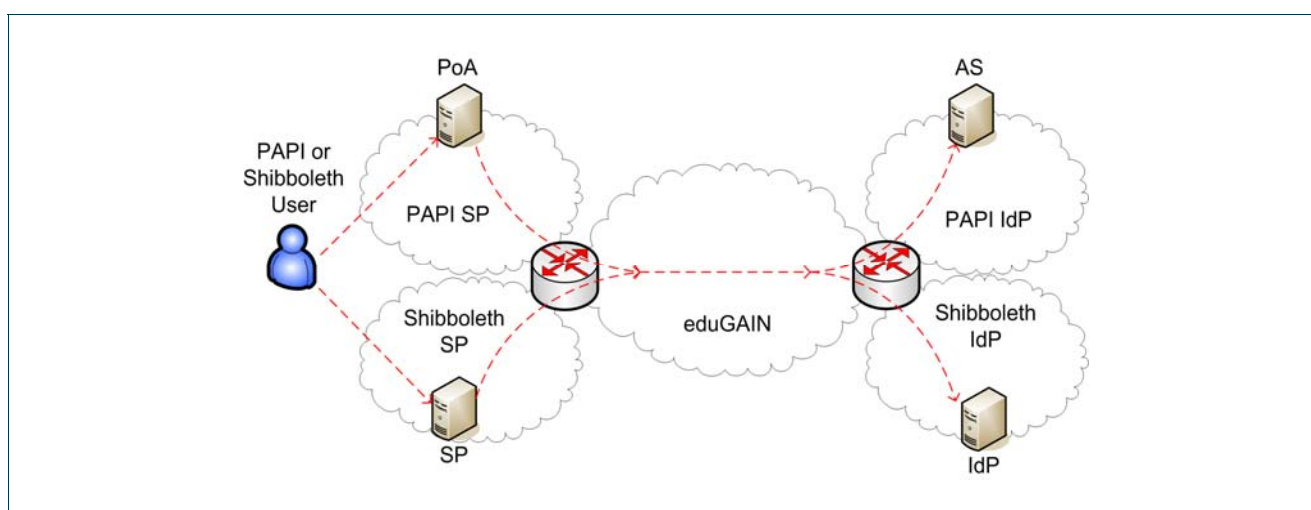


Figure 4.4. Different SPs and IdPs in DAMe

Next, two different kinds of federation middleware and the conversion from their specific authentication statements to and from the *eduToken* are described.

4.4.1 Shibboleth

Shibboleth [SHIB] is a web authentication and authorisation infrastructure (AAI) based on the use of SAML and web redirections to determine if a user can access a resource through their web browser. This decision is based on the user's information that is maintained in their home institution. This mechanism enables the

definition of identity federations in such a way that the user always authenticates to their home institution, and then the needed information is sent to where the request is made for authorisation. Shibboleth defines a Service Provider (SP) and an Identity Provider (IdP). The former is at the institution providing resources and the latter is at the institution managing the user's identity. In this way, when the user accesses some protected resource, the SP asks the IdP where the user is from, in order to obtain information about them. If the user was previously authenticated, the IdP returns the needed information, otherwise the user is redirected to the IdP to be authenticated. In this way, web SSO is provided.

Specifically, the authentication request that is sent to the Authentication Authority in Shibboleth is a URL-encoded message that contains the resource being accessed, the entity that must receive the response, or *shire*, and the identity of the provider. Also, the user name and password are included in the HTTP header.

The following is an example of Shibboleth authentication request:

```
https://idp.example.org/Shibboleth/SSO?target=network
&shire=https://sp.example.org/consumer
&providerId=provider.org
```

If the authentication is successful, the Authentication Authority sends a SAML Authentication Statement to the shire (= SP). This statement is similar to the following:

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  MajorVersion="1" MinorVersion="1"
  AssertionID="a75adf55-01d7-40cc-929f-dbd8372ebdfc"
  IssueInstant="2004-12-05T09:22:02Z"
  Issuer="https://idp.example.org/shibboleth">
  <saml:Conditions NotBefore="2004-12-05T09:17:02Z"
    NotOnOrAfter="2004-12-05T09:27:02Z">
    <saml:AudienceRestrictionCondition>
      <saml:Audience>http://sp.example.org/shibboleth</saml:Audience>
    </saml:AudienceRestrictionCondition>
  </saml:Conditions>
  <saml:AuthenticationStatement AuthenticationInstant="2004-12-05T09:22:00Z"
    AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
    <saml:Subject>
      <saml:NameIdentifier Format="urn:mace:shibboleth:1.0:nameIdentifier"
        NameQualifier="https://idp.example.org/shibboleth">
        3f7b3dcf-1674-4ecd-92c8-1544f346baf8
      </saml:NameIdentifier>
      <saml:SubjectConfirmation>
        <saml:ConfirmationMethod>
          urn:oasis:names:tc:SAML:1.0:cm:bearer
        </saml:ConfirmationMethod>
      </saml:SubjectConfirmation>
    </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

Figure 4.5 shows how the eduToken can be built based on the information included in the authentication response received from the Shibboleth IdP. It is necessary to fill in all the mandatory fields of the former with

information obtained from the latter. First, the *NotBefore* and *NotAfter* condition values can be obtained from the homonym fields in the *Condition* element of the authentication response. The *Result* field of the eduToken is set to *Accepted*, because the creation of the eduToken implies that the user is successfully authenticated. The most important field of the eduToken, that is the *SubjectHandle* that identifies the user in their home domains, is set to the SAML *Subject* element specified in the Shibboleth authentication response. Other specific information of the SAML elements of the eduToken, such as the *IssueInstant* or the *AuthnMethod*, can be obtained from the corresponding elements in the Shibboleth authentication response. Finally, although the *ProducerId* of the eduToken is included in the certificate related to the private key that was used to sign the token, the identifier of this entity is also specified in the *Issuer* field of the *Assertion* that defines the eduToken.

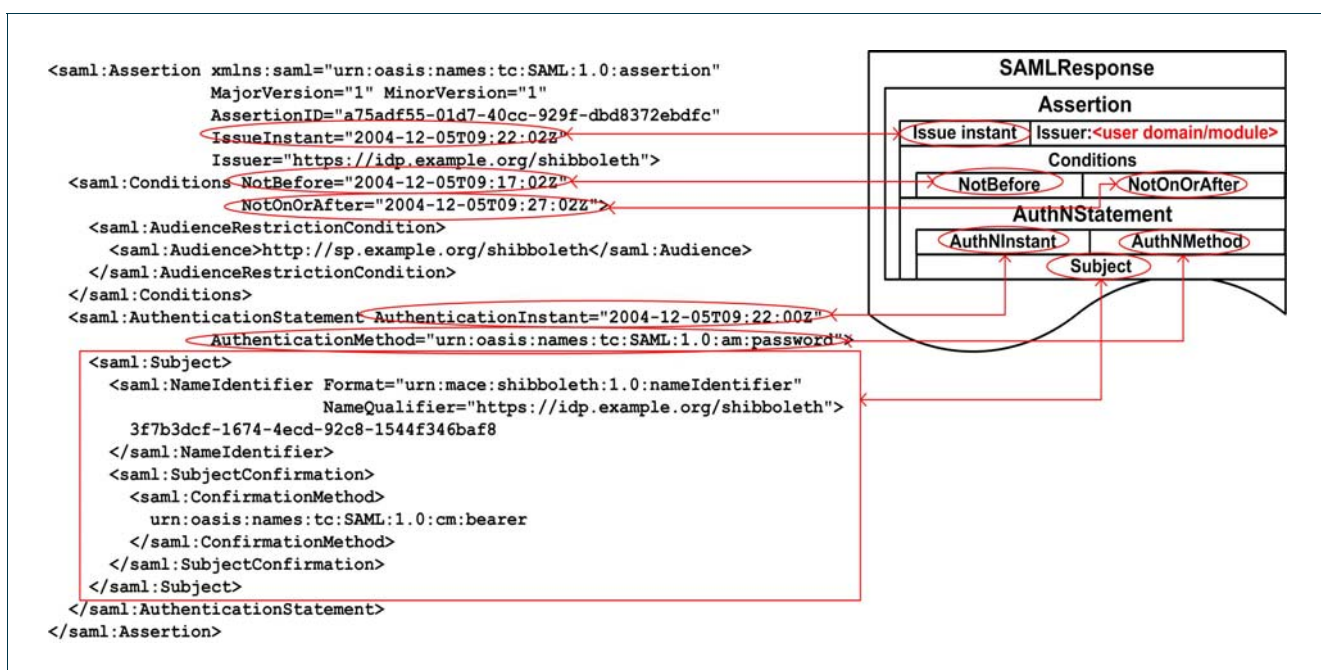


Figure 4.5. Conversion from the Shibboleth authentication assertion to the eduToken

4.4.2 PAPI

PAPI [PAPI] follows a different approach from Shibboleth because it does not use SAML. This system defines two kinds of entities:

- Authentication Service (AS).
- Point of Access (PoA).

First, the user authenticates in the AS and gets an authentication cookie named *PAPIAuthNcook*. The format of this cookie is *varList::nonce::TTL::clientAddress*, where:

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

- *varList* = a list of user attributes.
- *nonce* = a random value included for security reasons.
- *TTL* = the validity period of the cookie.
- *clientAddress* is the IP address of the user. This cookie is encrypted by the AS using a symmetric key in such a way that only the AS can decrypt it.

Furthermore, the AS returns to the user a list of URL-encoded authorisation requests for the configured PoAs. Each of these URLs is in the following form:

```
http://poa/AuthLocation?AS=ASId
&ACTION=LOGIN
&DATA=absc5...
&AURL=<accept page>
&RURL=<reject page>
```

Where:

- *ASId* = the identifier of the Authentication Service.
- *LOGIN* = the command to indicate the PoAs to perform the authorisation of the user.
- *DATA* = the user's assertions encrypted with the private key of the AS, which implies that these PoAs must have the public key of the AS.

The PoAs that authorise the user must send in the response two cookies:

- *Lcook* (format *timeStamp:location:serviceID:userData*).
- *Hcook* (format *isTemporary:userData:expiryTime:randomBlock:location:serviceID*).

These cookies are encrypted by the PoA using a symmetric key, in such a way that only the PoA can decrypt them. *Lcook* is a session cookie that is maintained in memory, while *Hcook* is a permanent cookie that maintains the state among different sessions. Using these two cookies, when a user that owns the appropriate cookies accesses a service protected by a specific PoA, the access is granted. If the user does not have the cookies, the PoA issues an attribute query and redirects the user to the AS:

```
http://ASURL?ATTREQ=PoAId&PAPIPOAREF=1179...&PAPIPOAURL=http://...
```

AS then authenticates the user either by using the *PAPIAuthNcook* or by requesting the user name and password. When the user is successfully authenticated, the AS redirects them to the PoA providing the assertions with the following URL, which includes the *DATA* field:

```
http://PAPIPOAURL?AS=ASId?ACTION=CHECKED&DATA=1a2b..
```

In a PAPI domain, the *eduToken* is built following a similar process to that in Shibboleth, but using the information that the PAPI AS provides to the BE. In PAPI, the authentication data about the user is included in the *PAPIAuthNcook*, but this information is only accessible by the AS, because the value of the cookie is encrypted using a symmetric key that only this entity owns. Therefore, to include the authentication information about the user in the *eduToken*, the encrypted data has to be included. Because the *PAPIAuthNcook* includes

the *nonce* and *clientAddress* fields, we can ensure that the value of this encrypted cookie is unique. Thus, the sequence of bytes that defines the encrypted value of the cookie can be used as a *SubjectHandle* for the eduToken. To complete the SAML *AuthNStatement* that defines the *SubjectHandle*, it is also necessary to set *password* as the *AuthNMethod*. The cookie also specifies its expiration time. Then, this information can be used to set the *NotAfter* element of the eduToken. The *NotBefore* field is set to the time the eduToken is being created. As Figure 4.6 shows, time constraints of the SAML statement are based both on the expiration time of the cookie and the creation time. The *Result* field is set to *Accepted* because the eduToken is only built when the user is successfully authenticated. Finally, although the *ProducerId* of the eduToken is included in the certificate related to the private key that was used to sign the token, the identifier of this entity is also specified in the *Issuer* field of the *Assertion* that defines the eduToken.

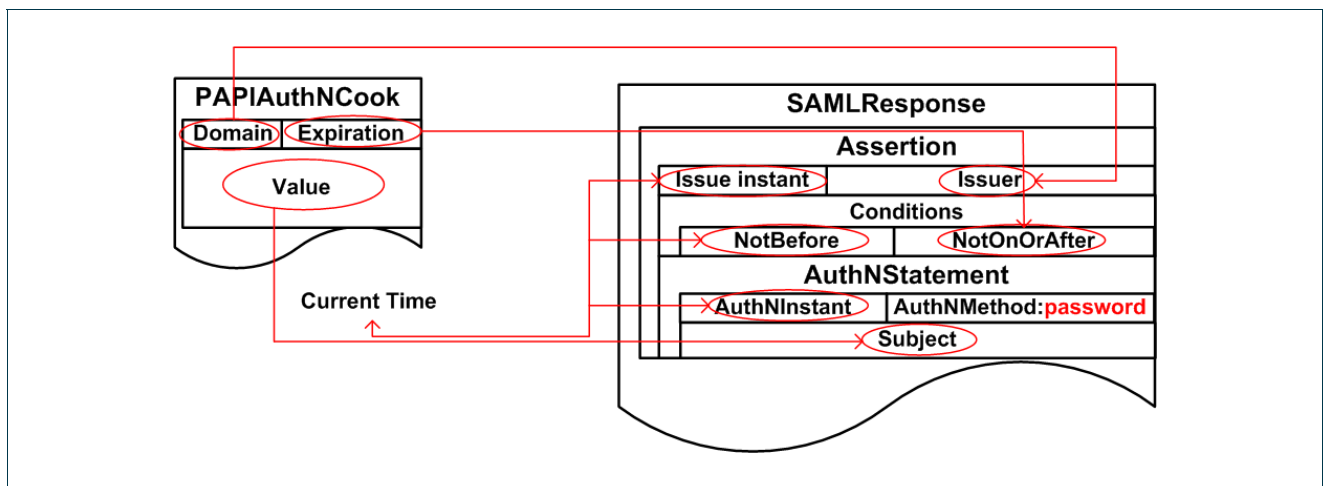


Figure 4.6. Conversion from PAPI User's Assertions to the eduToken

4.5 Data protection

In relation to data protection, the uSSO architecture follows an approach that is very similar to the one of eduroam [EDUSERV]. The uSSO architecture transports user related data on encrypted channels only, since we are using PEAP channels to distribute the eduTokens. The eduroam authentication process has not been modified, and therefore the use of an outer (anonymous) identity shall not disclose the user. The authentication credentials (i.e. passwords) are securely transported while they are traversing the eduroam infrastructure.

Additionally, as we will see in Section 5, the eduToken stays securely encrypted end-to-end between the identity provider and the user's personal device. An encryption algorithm will be even used to protect the token inside the terminal.

5 Unified SSO Client Middleware

For unified Single Sign On (uSSO) the DAME architecture uses specific middleware on the client, the end-user's device. This is different from the approach of AAls; for example Shibboleth only requires a browser, which comes pre-installed with current operating systems.

This means that during an actual deployment of the proposed architecture, special care should be taken to ensure that this middleware is distributed by the trusted home organisations to the end-users, preferably in a preconfigured way for installation to be as simple as possible. Although this could be considered to be a complication, it should be remembered that this situation isn't unique to the DAME uSSO architecture – for example, eduroam proposes supplicants that don't ship with many current operating systems and therefore have to be installed, and VPN clients are not pre-installed. Also, the client middleware used for common web-based SSO solutions (the browser) is often replaced by one not pre-installed, for a number of reasons. Therefore, this middleware distribution should not prove to be an issue.

5.1 Supplicant

In DAME an SSO-enabled supplicant is necessary for end users. The supplicant has to implement the 802.1x [8021X] protocol for the connection to the wireless network. IEEE 802.1x is a port based authentication protocol that can be used in any scenario where one can abstract out the notion of a port. It involves three types of roles in the authentication process:

- Supplicant.
- Authenticator.
- Authentication server.

Usually an access point acts as authenticator, while a wireless station is the supplicant that is authenticated by the RADIUS server. Several supplicants have been studied in order to select the most appropriate to incorporate the uSSO functionality.

Xsupplicant [XSUPPLICANT] is part of the Open1X project, an open source implementation of the IEEE 802.1X standard. This software allows a Linux, Windows, Mac OS or BSD workstation to authenticate with a RADIUS server using 802.1X and various EAP protocols. For example, it includes support for EAP-MD5, EAP-TLS,

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

EAP-PEAP, EAP-TTLS and others. The intended use is for computers with wireless LAN connections to complete a strong authentication before joining the network. Xsupplicant was chosen by the OpenSea Alliance, which is formed by leading vendors, such as Hewlett-Packard or Symantec, to develop and distribute an open source 802.1X supplicant.

On the other hand, WPA_supplicant is a WPA (WiFi Protected Access) Supplicant for Linux, BSD, Mac OS X and Windows with support for WPA and WPA2 (IEEE 802.11i / RSN). It is suitable for both desktop/laptop computers and embedded systems. It implements key negotiation with a WPA Authenticator and controls the roaming and IEEE 802.11 authentication/association of the WLAN driver. WPA_supplicant is designed to be a "daemon" program that runs in the background and acts as the backend component controlling the wireless connection. This supplicant includes 802.11i features such as WPA-PSK, WPA with EAP, or key management for CCMP, TKIP, WEP104 and WEP40. Besides this, it supports EAP-TLS, EAP-PEAP, EAP-TTLS and other EAP methods.

Finally, the SecureW2 EAP Suite is an Open Source EAP Framework that plugs in to the Microsoft 802.1X/EAP (EapHost) environment, available on all popular Windows platforms used today (Windows Mobile, Pocket PC, Windows 2000/XP, Vista). Currently the EAP Suite supports EAP-TTLS and EAP-GTC. Although it is an open source software, commercial support can be ordered for this product.

The selection of one of these alternatives to implement the supplicant for DAME was based on several requirements. It has to implement EAP-PEAP or EAP-TTLS and has to be open source to allow its modification. The uSSO architecture requires special features to be added to the supplicant, such as the recovery of the eduToken through the TLS tunnel, and therefore it is necessary to modify the source code of the supplicant. Another requirement is to be available for the most common operating systems. In that sense, SecureW2 can be discarded because it is only available for Windows. Finally, one important difference between xsupplicant and wpa_supplicant is the maturity of the code. Xsupplicant is on version 2 while wpa_supplicant is on version 0.5. Besides, the selection of Xsupplicant by the OpenSEA Alliance provides some guarantee for this software. For that reasons, Xsupplicant was the option selected to be used.

XSupplicant is the software of a client-side device involved in a handshake with an authenticator, which is located on the server-side. Both devices exchange authentication credentials using the Extensible Authentication Protocol, EAP [EAP], that offers different authentication methods such as PEAP [PEAP], EAP-TLS and EAP-TTLS. The authenticator, which is an access point, uses the RADIUS protocol to forward the credentials to an authentication server.

Among the methods available to authenticate wireless LAN connections, Protected EAP (PEAP) is possibly the most widely used. Rather than using digital certificates to authenticate users (with all the complexity that implies) PEAP can instead rely on a user password and a server certificate. Most organisations have standardised PEAP as the wireless authentication protocol of choice.

All the most commonly used 802.1X authentication protocols are based on TLS. TTLS and PEAP use TLS to authenticate the network to the user and provide security for the user-to-network authentication. XSupplicant uses the OpenSSL TLS implementation.

Between XSupplicant and the RADIUS server there is an exchange of messages. When the communication begins between them, RADIUS authenticates to the supplicant by sending its certificate. A PEAP tunnel is

established between the RADIUS server and the client for the authentication of the user to take place in a secure way. This will be based on the username and password, for example using the MsCHAPv2 authentication method.

The behaviour of the XSupplicant has to be modified in order to adapt it to some of the requirements of DAME. One of these changes is the support for packet fragmentation. XSupplicant only supports fragmentation in the handshake phase, but fragmentation is needed during the whole communication because the messages that are going to be sent are bigger than that XSupplicant expects. On the other hand the size of the buffers that uses XSupplicant needs to be changed because receiving larger messages means storing them in a larger buffer.

The packets received from the XSupplicant are encrypted using TLS. If the packet to be sent has to be fragmented, we need a way to indicate to the server that more fragments are expected. This type of packet is formed only by the EAP header (11 bytes of length) and is interpreted by the server as an acknowledgment for new fragments.

When the XSupplicant has the full packet, the token is extracted. Then it is verified that the TLV is of type VENDOR_SPECIFIC, the length of token is obtained and the token is read and stored.

New methods have to be implemented for reading the token that will be inside an EAP TLV included in the SUCCESS message. Figure 5.1 shows the token packet format. Also for this purpose, a new element has been included in the file configuration of the XSupplicant, which indicates where the token has to be stored.

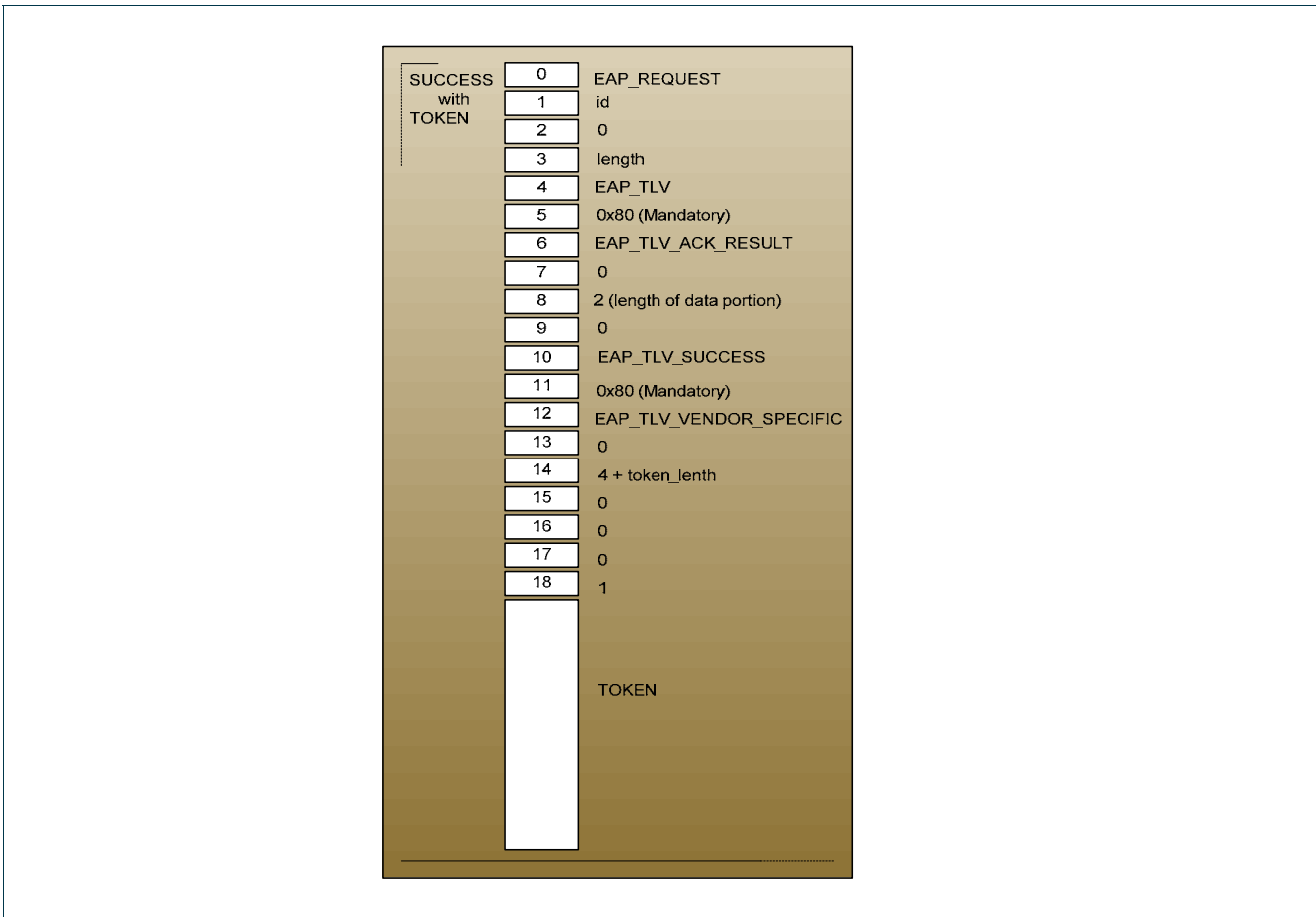


Figure 5.1: Token packet format

In order to carry out SSO, it is necessary for the XSuppliant to interoperate with the Token Manager. Therefore, a communication interface must be established between them so the token can be passed from the XSuppliant to the user's Token Manager.

5.2 Token Manager

To support the unified SSO architecture based on tokens as described above, a dedicated client for managing the eduTokens is required (the Token Manager). This is an application that is installed on end-user devices. Using a web browser (as in most web based AAls) is no longer sufficient, because the “signed on” state is now represented by files containing SAML Authentication Statements that are stored on the user’s device. Instead of the browser’s cookie store, a dedicated place for storing this sensitive information is used. The tokens are not stored among a lot of other, non-security related parts of information (as in the browser’s cookie store).

The eduToken is in itself a form of credential. With it, any service can be accessed that the rightful owner of the token is entitled to access. No further entry of a password is wanted, otherwise it would not be a SSO solution. Therefore it is essential that the tokens are always encrypted before storing them on the user’s device, or anyone using this device could steal that token and impersonate the rightful owner. Additionally, all eduTokens

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

must be signed using valid certificates from a source that is trusted by the federation, so no modification is possible either of the validity, scope, or the issuer of the token.

The primary purpose of the Token Manager is to link network authentication to web authentication. To achieve this purpose, the Token Manager takes the eduToken from the supplicant that received it during the network authentication phase. It then provides an interface for web components to request the token when authentication is enforced, so the eduToken can be sent instead of requiring the user to enter their username and password. Service Providers protecting web resources need not be modified but must be extended by a component that supports the token profile and can issue such requests.

To reduce the burden to the end-users, the installation effort for the Token Manager must be minimal. The current solution is a Java application that only requires the Java runtime to be installed; there is no requirement on the operating system or middleware.

One advantage of having a dedicated software client for uSSO is that the end-user can decide to be informed about what is happening (for example, what is the “signed on” state, what digital identity they are currently using and giving to the Service Providers, and possibly in future developments they might have control over what attributes are sent). But those users not wishing to see such information get an improved experience as well – the Token Manager only needs to be running in the background, and by providing unified SSO less typing of passwords is required.

Another advantage of having a dedicated software client for uSSO is flexibility. The “signed on” state can be displayed to the user, making them aware of the established security context. With this client, full SAML assertions can be used as tokens. Also the tokens would not need to be stored among a lot of other, non-security related parts of information (as is the case with the browser’s cookie cache). Furthermore, such a client is valuable for accessing non-web-based services. The following sections contain specifications for this dedicated client.

Yet another advantage is that a number of features become enabled (or at least simpler to implement) by having a dedicated software client. Some possible examples are:

- eduTokens containing user attributes in addition (or instead) of the user’s ID.
- Support for network access control (NAC) by “posture(*)-tokens”.
- Support for further communication protocols and/or further service types, e.g. network access via token: A token is pre-loaded to the client that is valid for a longer period of time (e.g. a few days). During the network authentication phase the token is used instead of other credentials.
- Support for multiple identities/profiles and federations: Users may want to use one of several virtual identities they have, or decide what profile to use for a service provider, or even a single transaction. Furthermore, users would possibly have accounts in more than one federation, and be signed onto more than one of them at any one time.

(*) The term “posture” refers to one part of a Network Admission Control (NAC) system and means “security and policy compliance information about the end-users device” which is sent from trusted modules on that device.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

6 uSSO Profiles

6.1 Authorisation for Network Properties

As discussed previously, one of the target scenarios for uSSO is the integration of authorisation decisions into the eduroam infrastructure. To extend the current eduroam infrastructure to offer authorisation mechanisms, with minimum impact on the institutions willing to continue using the standard process, it has been decided to preserve the current eduroam authentication mechanism in the implemented approach but to deploy a new authorisation infrastructure. In this way, using this alternative, institutions can decide whether they want to use an extended RADIUS server connected to the NAS-SAML Policy Decision Point (authentication and authorisation), or a standard one (only authentication).

Using this approach, once a roaming user is authenticated following the standard eduroam mechanism, the target RADIUS server uses NAS-SAML to carry out the authorisation process (Figure 6.1). Therefore, the home institution is consulted about the user's attributes and, once this information is recovered, an authorisation decision is obtained using the PDP. Moreover, a set of obligations can be returned along with the authorisation decision containing some specific network properties to be enforced by the access point.

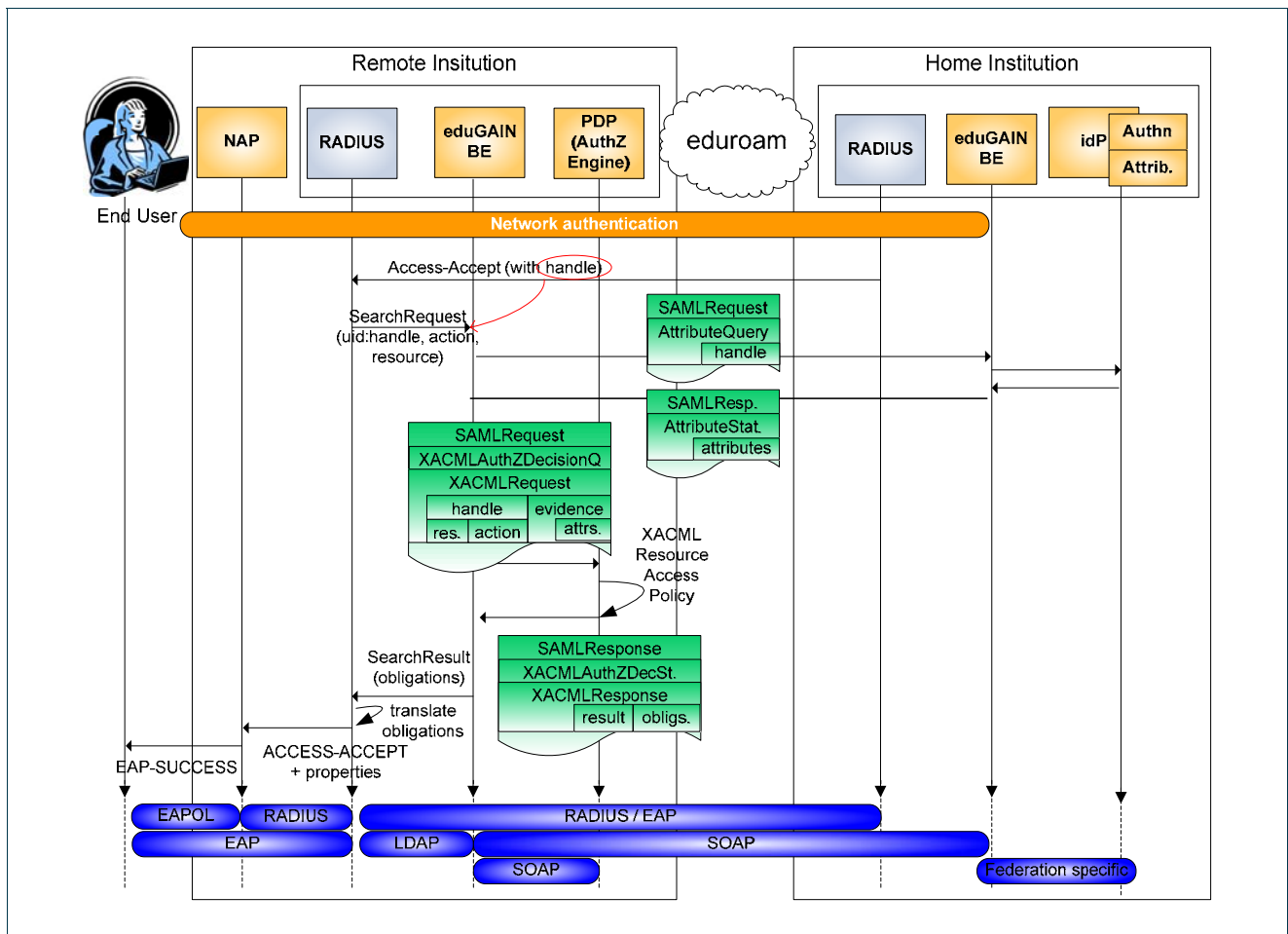


Figure 6.1. Network authorisation

The RADIUS server will ask for the network properties or obligations to be applied to the user's connection, which are defined by the user attributes. The RADIUS server delegates the process of obtaining the user attributes and checking the properties to be applied to the local BE, as follows:

- The RADIUS server sets off this process by means of a LDAP connection to the BE. It sends an LDAP *SearchRequest* message, including the authentication handle, the target resource (*network*) and the required action (*access*). Then, the remote BE sends a SAML *AttributeQuery* message (as described in eduGAIN) to the home BE including all this information. It is worth noting that the location of the home BE could be obtained either from the handle or from the MDS service, as proposed by eduGAIN.
- The home BE, after receiving the query, forwards it to the corresponding attribute authority, in this case the IdP. As before, the communication protocol between BE and IdP depends on the implemented technology.
- The IdP, by means of the handle, selects the user attributes and, optionally, could use an attribute release policy to check which ones can be revealed to the remote institution. Finally, it generates a SAML *AttributeStatement* including the selected attributes, which is sent back to the remote BE.
- Once in the remote institution, the BE, using the NAS-SAML PDP entity, sends an authorisation decision query in order to know the set of properties that have to be applied to the user network

connection. This query includes the user subject and attributes, the target resource and the required action. The communication between the BE and PDP is made by means of SAML *AuthorizationDecisionQuery/Statement* messages, transported over SOAP, as described in the NAS-SAML proposal [NAS].

The authorisation decision process conforms to a *Resource Access Policy*, described in [POLI], which includes the set of network properties as XACML obligations. Once these properties have been selected, they are forwarded to the RADIUS server by means of a LDAP *SearchResult* message, which will be the responsible for enforcing them into the AP. Examples of those properties are the *Session-Timeout*, *MaxBandwidth*, or *VLAN-ID*.

6.2 Profile for Token-Based Web Authentication

In this section a new profile for token-based web authentication is specified. A number of components are involved, though not necessarily an Identity Provider (which would only be needed for requesting attributes, not for the authentication itself). This profile is not (yet) supported by existing Service Provider (SP) implementations themselves. Instead, the remote eduGAIN bridging element (r-BE) is modified, as there are far fewer installations of BEs than SPs that would need to be updated. In case of a central BE it would be only one per federation.

6.2.1 Architectural Components

Of the following components, two of them are modified (supplicant and remote Bridging Element) and two are new (Token Manager and Token Fetcher).

- **Network Access Point (NAP):** A hardware device providing wireless access to the network. It enforces usages of EAP for 802.1X authentication. It is next to a RADIUS server that is the entry point to the eduroam infrastructure.
- uSSO Client:
 - **Supplicant:** The user's supplicant that is used to connect to eduroam, using the Extensible Authentication Protocol (EAP) with any EAP-type (EAP-TTLS or Protected EAP, for example). It is extended to receive a token, and give it to a SSO client (token client).
 - **Token Manager:** Middleware installed on the user's device, required for uSSO. Receives tokens from the supplicant, manages and securely stores them. Responds to token requests from a JavaApplet.
- **Browser:** Standard user software to access web sites and services. JavaApplets can be executed within, the Java Plug-in must be installed.
- **Service Provider (SP):** A service provider that offers any kind of web based services and controls access to them. It is part of a federation and trusts identity providers (remote BEs) to provide user authentication and attributes.
- **WAYF:** A component within a federation that allows users to enter information about where to find the responsible Authentication Authority, and that redirects AuthenticationRequests from an SP to the selected Authentication Authority. Usually that Authentication Authority is an IdP at the home domain of the user, in case of token-based uSSO it is the uSSO client.

- **Remote Bridging Element (r-BE):** eduGAIN-aware component that bridges one federation to others. To an SP it acts as an IdP. Using the eduGAIN infrastructure it can request user attributes for authorisation. It is extended to request an eduToken from the uSSO client, and to understand and validate that token. There can be one BE for the whole federation, or one BE specific to that service provider.
- **Token Fetcher:** A signed JavaApplet that is sent to the user and executed within the browser. It requests the token and uses HTTP POST via an HTML form to send the token to the r-BE.

6.2.2 Message Flow

The message flow is shown in Figure 6.2:

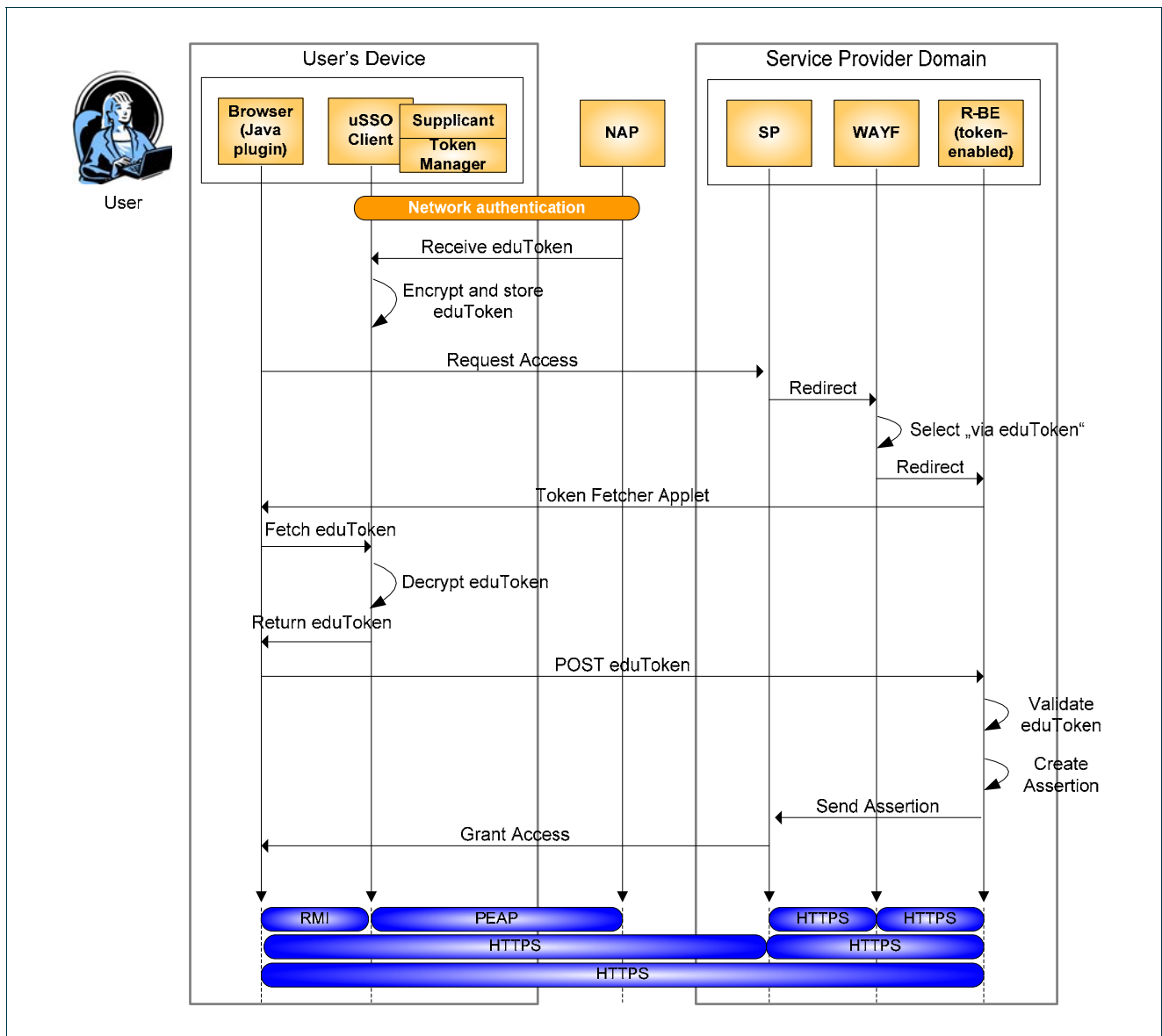


Figure 6.2. Token-Based Web Authentication

The steps are as follows:

1. The token is received during the network authentication phase, inside an EAP TLV.
2. The token is encrypted and stored by the uSSO client.

Note: Nothing happens until the user tries to access a SP-protected web resource.

3. The user tries to access a SP-protected web resource.
4. This request is redirected to a WAYF (this step is optional)
5. The user selects “via eduToken”. Otherwise normal eduGAIN Web SSO would be done.
6. The request is redirected to the modified eduGAIN r-BE.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

7. The r-BE requests the eduToken via the browser from the uSSO client. A signed Java Applet is sent inside an HTML page sent to the user's browser, executed therein, and requests the eduToken from the uSSO client.
8. The eduToken is decrypted by the uSSO client.
9. The eduToken is given to the applet and sent in a reply back to the r-BE. If no token can be found, an error code is sent.
10. The r-BE validates the eduToken (checking conditions plus signature).
11. The r-BE creates an assertion from the eduToken. This assertion is in a federation-specific format (e.g. Shibboleth or PAPI).
12. The assertion is given to the SP. In case of an error or a failed validation, the error is given in some federation-specific form.
13. The SP grants access to the user.

6.3 Unified SSO and Authorisation for Grid Computing

Grid computing is widely used as a tool to support applications with especially high requirements for computational power, storage space, network speed, and more. These applications cannot be executed on a single computer but need a distributed environment with a large number of often specific hardware devices to get results. This is common in scientific fields such as physics, medicine or astronomy. Projects in such a context make high demands on their authentication and authorisation components because of the distributed and dynamic nature of these organisations, the complexity of the technologies involved, and the security requirements (for example in the medical context hospitals must protect individual patient information).

The aim of this part of DAME is to demonstrate the unified SSO and the authorisation capabilities of the two parts described above by applying them in a Grid computing scenario. Some adaptations to Grid specific technologies are needed and described, but all parts of the DAME infrastructure can be reused without requiring fundamental changes. This Grid scenario shows that DAME can provide a federated AAI for Grid research projects, thereby allowing them to connect to the European eduroam and eduGAIN confederations. The second benefit is that only by implementing a concrete and complex scenario as the chosen one, the full capabilities of the DAME architecture can be highlighted.

6.3.1 Architectural Components

Figure 6.3 shows the architectural components required for the complete grid scenario. The elements in blue are Grid specific and added to non-grid specific parts in orange. In the picture the blue arrows show the authentication phase (using the eduToken) and the red arrows show the authorisation phase (via eduGAIN). The network authentication elements are not shown.

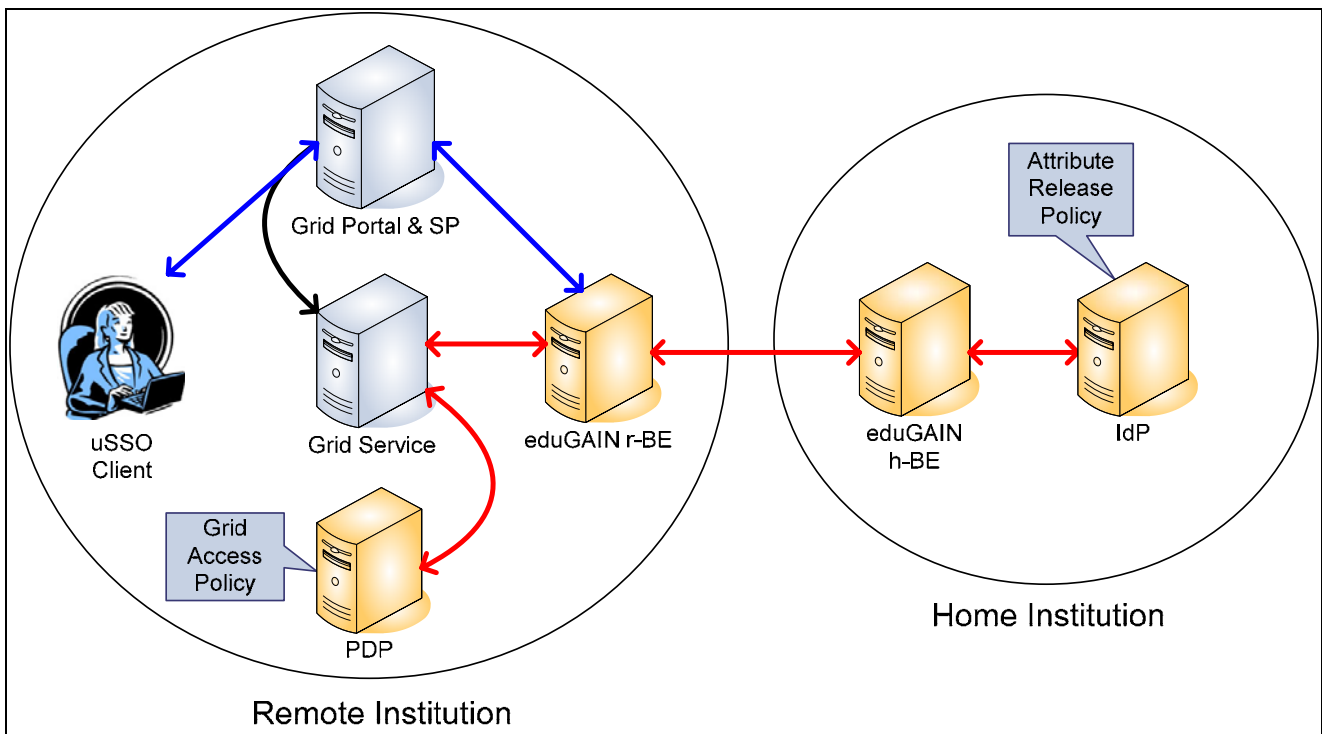


Figure 6.3: DAME Components of the Grid Scenario

The following are new components:

- **Grid Portal & SP:** This component consists of a Grid Portal that has been modified to work with the adjacent Service Provider (SP), taking the place of the usual web resource. It provides an easy-to-use interface to end-users and enforces federated access control by the SP. The portal contains a number of portlets that take the role of Grid clients and interact with the underlying Grid Service(s).
- **Grid Service:** The Grid Service is a statefull web service that has been implemented with the Globus Toolkit following the WSRF specification. End-users access it through grid portlets of the Grid Portal. The Grid service enforces authorisation for the operations that grid portlets want to perform. It requests the user's attributes by contacting the eduGAIN r-BE, and then uses them as input to an authorisation request to the PDP.

6.3.2 Message Flow

An overview of the complete grid scenario is:

- A user accesses the eduroam network and receives a valid eduToken on his uSSO client.
- When the user requests access to the Grid portal, the eduToken is used and the uSSO authentication is performed.
- If the authentication is successful, the Grid portal enables the user to select any of the Grid portlets that are available to them.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

- On selecting one grid portlet, a grid request together with some user information is sent to the grid service. The grid service obtains the user's attributes and then requests authorisation from the PDP.

The following sections and diagrams show the flow of messages, with the sequence being divided into two for a clearer picture.

6.3.2.1 Authentication via eduToken

Figure 6.4 represents the authentication process:

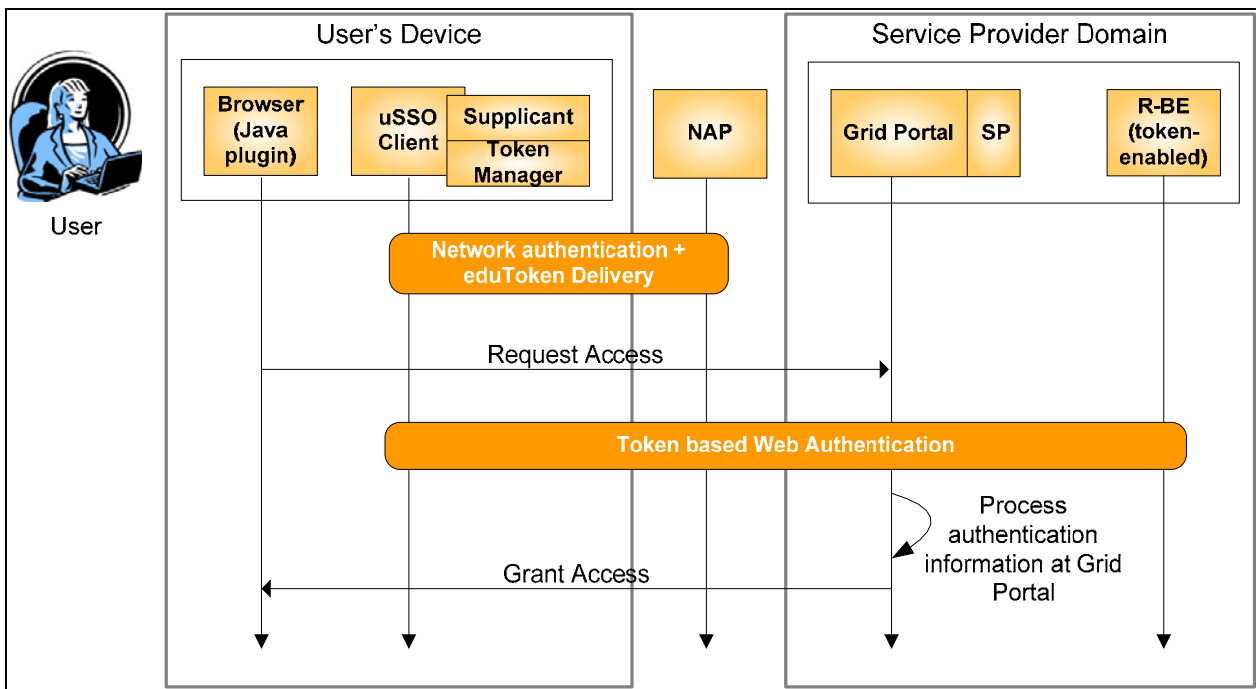


Figure 6.4: Grid Authentication with eduToken

1. Network authentication via eduroam, a valid eduToken is delivered.
2. Request Access: The user selects the DAME login at the Grid portal.
3. The Service Provider enforces federated authentication and redirects to a WAYF. Token-based web authentication is performed by fetching the eduToken from the uSSO client to the token-enabled remote BE.
4. Process user's information: When the Token based Web authentication has finished with success, the SP lets the protected resource (in this case, the gridsphere portal) be accessible. At this point, the Gridsphere login portal captures DAME user details that the Service Provider has received for that session (such as the user handle and the IdP) and processes them in order to make them available for the Gridsphere application.
5. Access is granted and the Grid portal shows the grid portlets available for this user.

6.3.2.2 Grid Authorisation

This process happens when the user has been already authenticated at the Grid portal via the Token based Web profile. Figure 6.5 represents the authorisation process:

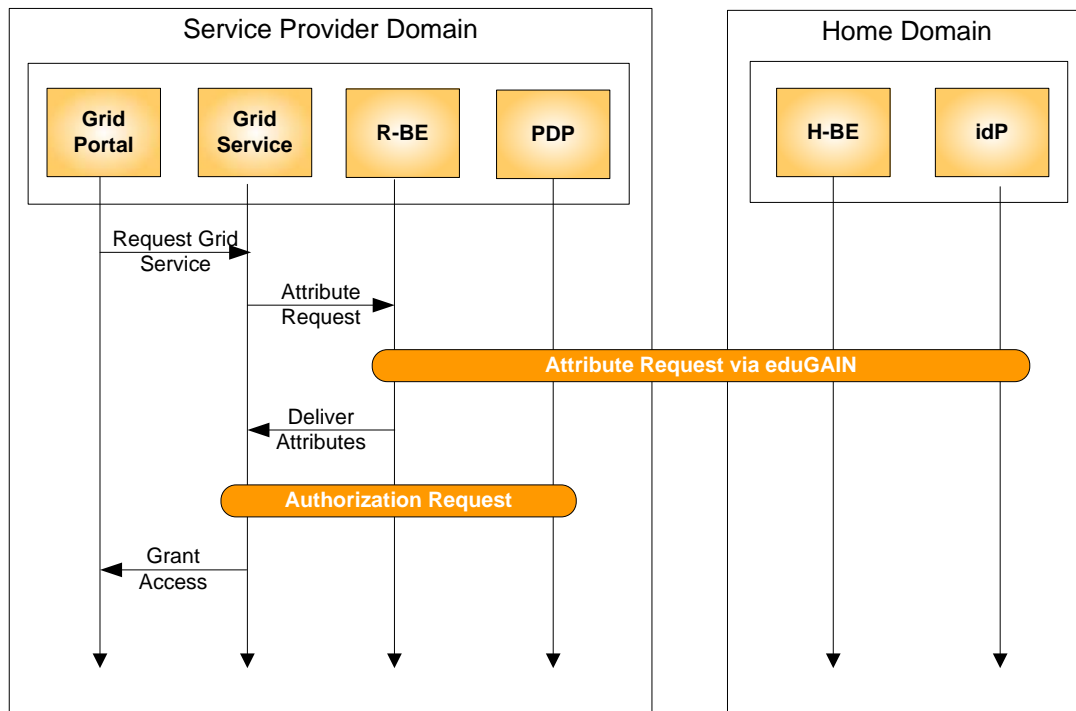


Figure 6.5: Grid Authorisation

1. **Request Grid Service:** The user selects one of the available grid portlets to execute. The portlet gets the user's information and connects to the grid service passing the user's information as parameters.
2. **Attribute Request:** The grid service performs an attribute request based on the parameters passed by the grid portlet, sending an attribute request to the remote BE. Now the process to obtain the attributes is the same as in the usual eduGAIN attribute request profile.
3. **Authorisation Request:** Upon receiving the attributes the Grid Service contacts the PDP in order to obtain the authorisation decision. The PDP, based on the policies of the Grid Service, will make that decision and send it back to the Grid Service.
4. **Grant Access:** If the authorisation of the user to that grid service is successful, the grid portlet connects to the grid service and its task is executed.

7 Validation

This section describes the testbed that has been deployed to validate the architecture described above.

7.1 Description of the testbed

The testbed deployed implements the architecture and profiles designed. Specifically, the intention of this testbed is to check the following points of the architecture:

- Initial network authentication based on eduroam.
- Request of the authentication assertion to the IdP by the home RADIUS server, via Home Bridging Element (Home BE).
- Generation of the eduToken at the Home BE.
- Delivery of the eduToken to the user.
- Secure storage of the eduToken.
- Encapsulation of the handle in a RADIUS attribute to be used later by the remote RADIUS server.
- Use of this handle to get the appropriate network properties for the user via LDAP.
- Delivery of the user's attributes from his home to use them to determine the network properties by means of XACML policies.
- Enforcement of the network properties in the Access Point for the user.
- Use of the eduGAIN MDS to discover the URL of the BE related to the user's home institution.
- Use of certificates issued by the eduGAIN PKI.

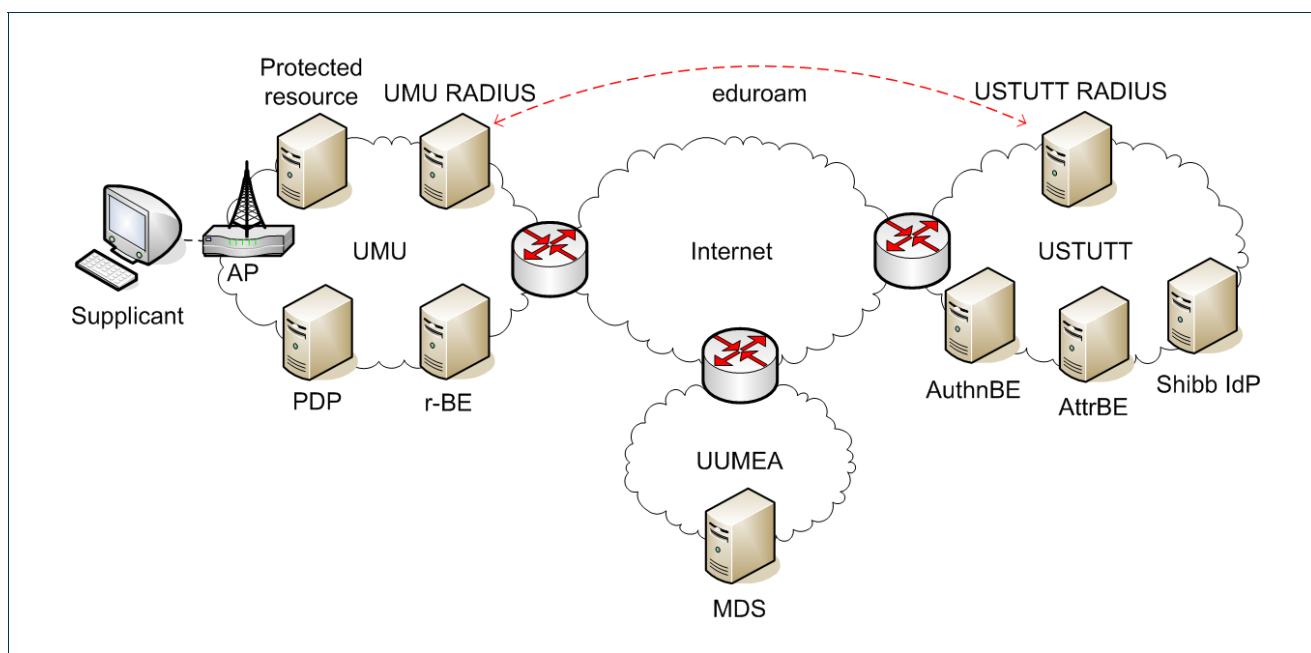


Figure 7.1: uSSO testbed

There is a direct translation between the architecture developed and the elements of the testbed, shown in Figure 7.1. The scenario for the testbed is made up by three institutions: University of Murcia (UMU), University of Stuttgart (USTUTT) and University of Umea (UUMEA). UMU acts as the Remote Institution where the roaming user is trying to access to the network. USTUTT acts as the Home Institution the user belongs to. Finally, UUMEA is the institution that holds the Metadata Service (MDS) for the federation. Furthermore, the Radius servers from UMU and USTUTT are connected through the eduroam hierarchy.

The description of each element is as follows:

- **Suppliant.** This is the software used by the client to connect to the network. It is an 802.1X client software, but extended with the specific features described previously in this document. These features include the reception of the eduToken through the PEAP tunnel established in the authentication process. Later, the eduToken is encrypted and stored in a file. This encryption is based on the AES 128 algorithm using a password.
- **AP.** This is the access point that provides wireless access to Internet to the user. Also it is where the network properties will be enforced. Currently, only the session-timeout property is enforced in the AP.
- **UMU Radius.** This element is the Radius server of the remote institution. It is responsible for redirecting the user authentication request to the appropriate Radius server through the eduroam hierarchy and for providing the network properties to enforce in the AP.
- **RemoteBE.** This is the eduGAIN Bridging Element that connects UMU to the federation. This BE is in charge of obtaining the user's attributes from their home institution and contacting the PDP to obtain the appropriate network properties for that user. To find the user's home institution, this element consults the MDS to get the URL of the proper AttrBE.
- **MDS.** This element maintains the metadata of the federation.

- PDP. This is an XACML PDP that receives an SAML Authorisation Decision Query from the RemoteBE and checks that query against the XACML policies defined on the system, determining the network properties for the user.
- USTUTT Radius. This element is the Radius server of the user's home institution. It receives the authentication request from the eduroam hierarchy, authenticates the user locally, and sends back the response. Also, if the user is successfully authenticated this element obtains the eduToken from the AuthnBE, which is sent to the user through the PEAP tunnel established during the authentication process.
- AuthnBE. This BE receives authentication requests from the home Radius server, and forwards it to the IdP in the appropriate format. Then, when the response from the IdP is received, it builds the eduToken based on that response, which is returned to the Radius server.
- AttrBE. This BE receives attribute queries from the RemoteBE, which are forwarded to the IdP in an appropriate format. Later, when the requested attributes are received from the IdP, they are sent back to the RemoteBE.
- Shibb IdP. The IdP is the central entity that manages the user's identity and attributes. It is implemented using Shibboleth. This entity receives authentication queries from the AuthnBE and, if the user exists in this IdP, an authentication response is sent back to the BE. It is important to note that, in this scenario, the IdP is not authenticating the users (only the home Radius server is performing this task). The IdP only generates the appropriate authentication response that will be used to build the eduToken. The IdP also receives attribute requests from the AttrBE. In that case, if the user's handle included in the request is valid, the attributes associated with that user are returned.

To deploy the testbed without modifying the current eduroam infrastructure, a new level of RADIUS servers has been added to the hierarchy. In this way, as Figure 7.2 shows, the RADIUS servers used in this work are located under the institutional RADIUS in UMU and USTUTT. Therefore, we have to take into account that the time measurements obtained for eduroam include an additional level of RADIUS servers. In a production environment, this additional level might not be necessary.

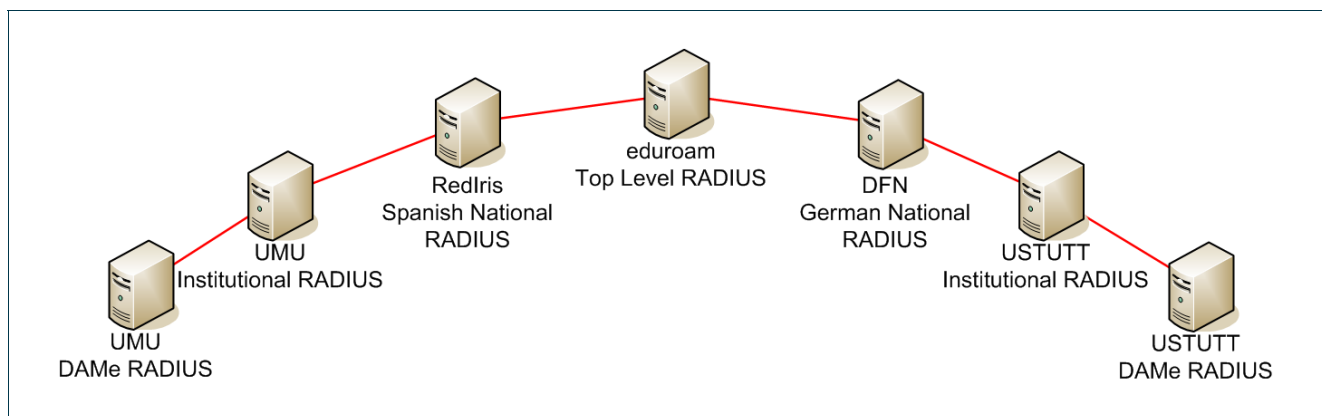


Figure 7.2: Extended eduroam hierarchy

Finally, the details of the hardware and software elements used to deploy the testbed are shown in Table 1.

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

Element	Hardware	Operating System	Base Software	Extra libraries
Supplicant	Pentium D 3 GHz 1024 MB RAM	Ubuntu Linux Kernel 2.6.22	XSupplicant 1.2.8	OpenSSL 0.9.8a
AP	Avaya AP-3			
UMU Radius	Pentium IV 3 GHz 512 MB RAM	Ubuntu Linux Kernel 2.6.15	Freeradius 1.1.3	
Remote BE	Pentium IV 3 GHz 1024 MB RAM	Ubuntu Linux Kernel 2.6.22	Tomcat 5.5.12 Penrose 1.3.1	eduGAIN 0.6 openSAML 1.1 openSAML 2.0
PDP	Pentium IV 3 GHz 1024 MB RAM	Ubuntu Linux Kernel 2.6.22	Tomcat 5.5.12	openSAML 1.1. sunXACML 1.2
USTUTT Radius	Pentium IV 3 GHz 1024 MB RAM	Ubuntu Linux Kernel 2.6.15	Freeradius 1.1.4	
Shibboleth IdP	Pentium IV 3 GHz 1024 MB RAM	Windows 2003 SE Service Pack 2	Shibboleth IdP 1.3	
Authn BE	Pentium IV 3 GHz 1024 MB RAM	Windows 2003 SE Service Pack 2	Tomcat 5.5.23	eduGAIN 0.5 openSAML 1.1 httpClient 3.0 htmlParse 1.6
Attr BE	Pentium IV 3 GHz 1024 MB RAM	Windows 2003 SE Service Pack 2	Tomcat 5.5.23	eduGAIN 0.5 openSAML 1.1 httpClient 3.0

Table 1: Hardware and Software details

7.2 Initial authentication and uSSO token delivery

To demonstrate that the architecture developed is feasible, several tests have been performed over the testbed described above. In these tests, time measurements have been taken to control the time that each part of the process consumes. Then, the time results obtained in the uSSO testbed have been compared with the time that eduroam takes to authenticate a user. The specific measurements taken in the testbed are the following:

- eduroam. This is the time that the authentication process takes in the standard eduroam. It is the starting point of this analysis since it represents the current infrastructure. However, we must take into account the new level added in the eduroam hierarchy.
- Extended authentication. It is the time since the user runs the client to connect to the network until the connection is available. Besides it includes the generation of the uSSO token and its delivery to the user.

Apart from these final results, we have also analysed the influence of the different phases during the extended authentication:

- Shibboleth authentication. This is the time that the IdP needs to validate the user and to generate the authentication assertion needed to build the uSSO token. It is measured in the AuthnBE.
- Token delivery. This is the time that the basic eduroam authentication process is incremented due to the delivery of the uSSO token. It cannot be calculated exactly because the token delivery is part of the

authentication process, and therefore cannot be separated. But we can get an estimation if we subtract the *Shibboleth authentication* time and the eduroam time from the *Extended authentication* time.

Both tests (the first based on eduroam and the other based on the testbed) were executed 105 times sequentially. The time results from the former are shown in Figure 7.3, while the time results for the latter are shown in Figure 7.4. In these illustrations the time is expressed in milliseconds, and based on the percentile distribution.

These graphics allow us to see how approximately the 95% of the values for each time measurement are uniform. Only 5% of the results are different. These values could be due to some punctual problem in the network or in the computer where the entity is running. Because the majority of the values are uniform, a median average is best suited as a representative measure. The medians of all the time measures taken are summarised in the following table. These values will be used as the representative value of each time measurement.

eduroam	Extended authentication	Shibboleth authentication	Token delivery
1568	2817	500	736

Table 2: Median of the values (in milliseconds)

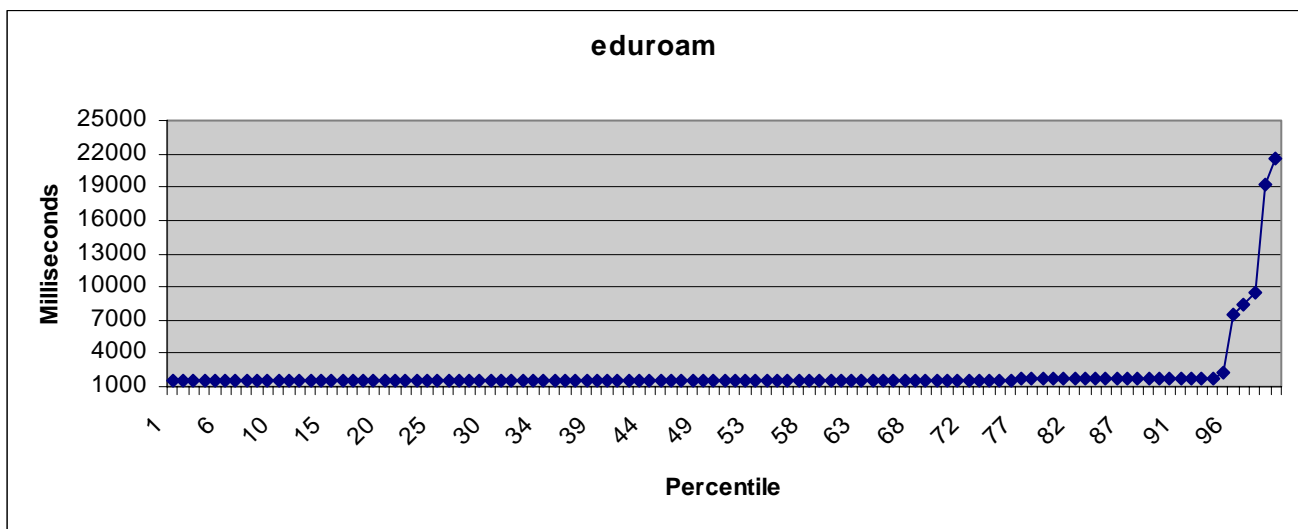


Figure 7.3: Time analysis. eduroam

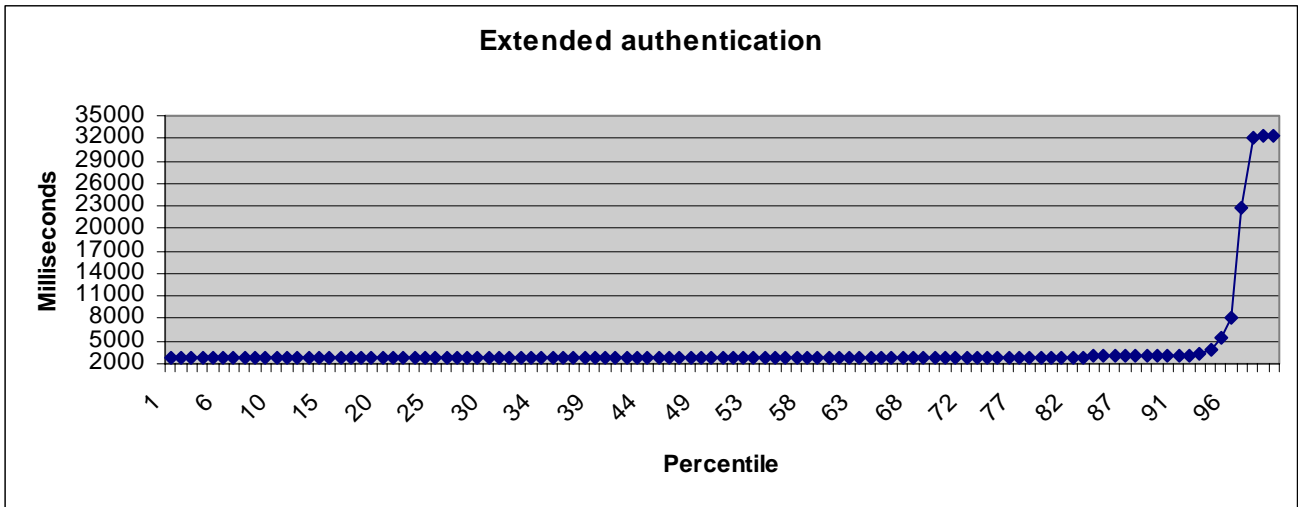


Figure 7.4: Time analysis. Extended authentication

From these figures we can see that the time that the user has to wait until the network connection is available is 2817 ms, less than 3 seconds. As Figure 7.5 shows, this time includes the token delivery and the Shibboleth authentication, which represents an increase around the 78% over the basic eduroam authentication process. However, this is an initial step that it is going to be carried out only once at the beginning. Therefore it is a reasonable time for the user to be waiting for the network connection. Also, we have to take into account the benefit derived from the SSO mechanisms. From now on, access to services that support the use of the uSSO tokens is immediate, while the use of the standard eduroam requires further re-authentication.

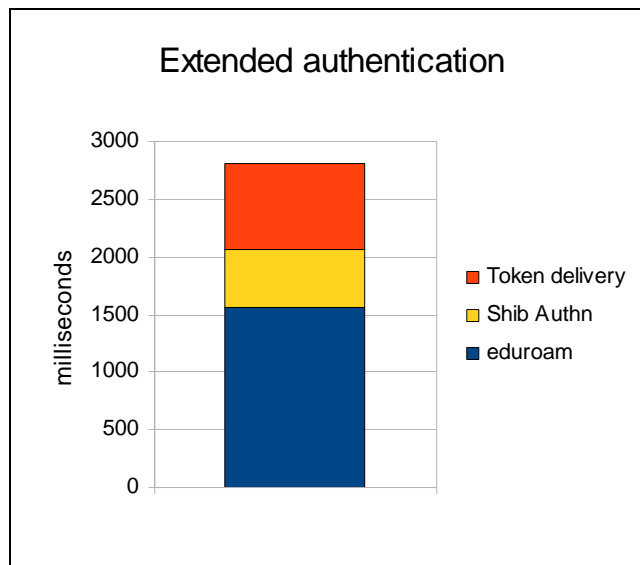


Figure 7.5: Extended authentication time components

7.3 Profile for network authorisation

This profile allows the remote institution to enforce specific network properties for the user that has authenticated using the already described architecture. The increase of the time that the user has to wait until the connection is available due to the network authorisation phase is studied next. The time measurements taken in the testbed are:

- MDS query: The time needed to obtain the URL of the AttrBE from the MDS.
- Attribute request: The time that the RemoteBE has to wait to get the user's attributes from the IdP.
- Shibboleth attribute request: The time that the IdP takes to recover and return the user's attributes.
- PDP query: The time since the RemoteBE sends the user's attributes to the PDP until it receives the network parameters.

Based on these measured values, the following times were calculated:

- Network authorisation: The time necessary to obtain the user's attributes and, based on them, determine the proper network properties. This is calculated as the sum of the *MDS query* time, the *Attribute request* time and the *PDP query* time ($MDS\ query + Attribute\ request + PDP\ query$).
- eduGAIN request: The time that the eduGAIN protocol needs to transport the attribute request and attribute response between the remote and home BEs. This is calculated as *Attribute request* time minus the *Shibboleth attribute request* time ($Attribute\ request - Shibboleth\ attribute\ request$).
- Authn & authz: The total time needed to establish the connection for a user in a remote institution that is also enforcing specific network properties for the user. This time includes the *Extended authentication* time and the *Network authorisation* time ($Extended\ authentication + Network\ authorisation$).

The medians of these values, calculated after the execution of the test 105 times sequentially, are summarised in the following table. The time to gain access to the network, as Figure 7.6a) shows, goes up until 3376 ms. This time represents an increase of the 18,9% over the extended authentication and 112,6% over the basic eduroam authentication. But still, due to it is necessary to carry out this process only once, less than 3,5 seconds is not a long time to wait for the user to get the initial network access.

MDS query	eduGAIN request	Shibboleth attr. request	Attribute request	PDP query	Network authorisation	Authn & authz
388	87	31	110	30	531	3376

Table 3: Medians of the network authz values (in milliseconds)

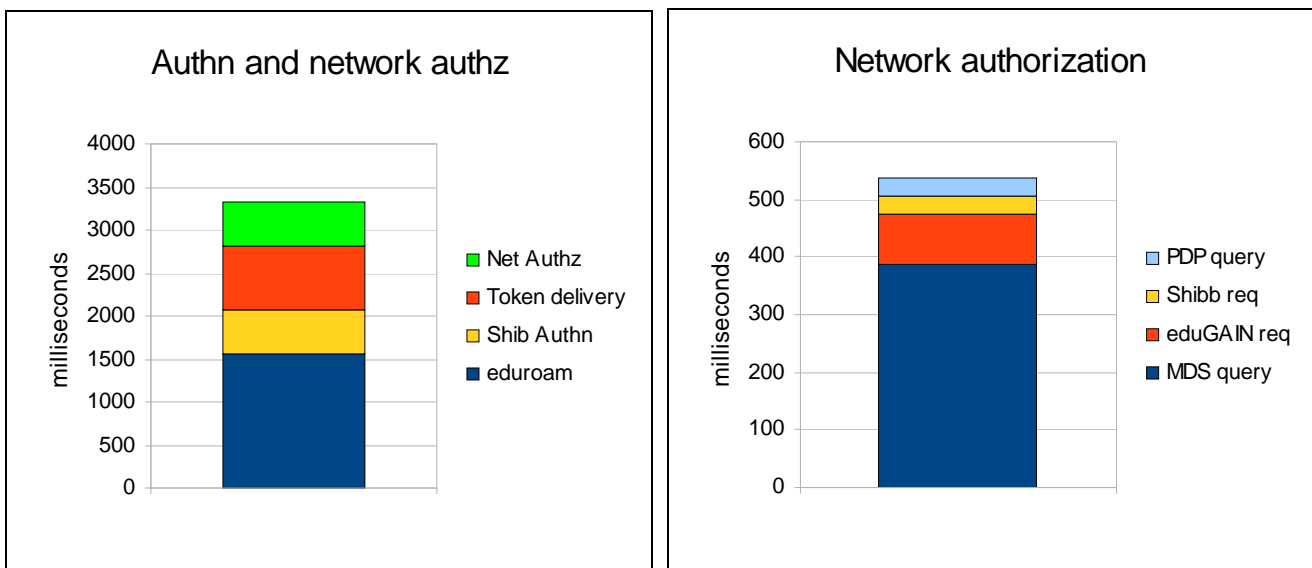


Figure 7.6: a) Global time b) Network authz time components

Figure 7.6b) breaks down the different components involved in the authorisation phase. We can see how the time needed to query the MDS represents the 72,3% of the network authorisation time. Therefore, if the time needed to access this eduGAIN component is reduced, the whole authorisation time can be reduced substantially. In the testbed, the *MDS query* time (388 ms) includes the HTTP request and response from UMU to UUMEA, and the time that the MDS needs to process the request. Comparing this time with the time needed by other HTTP communications over the network in the testbed, for example the *eduGAIN request* time (87 ms) that represents a SOAP/HTTP request and response from UMU to USTUTT, the majority of this time appears to be due to the MDS processing. At this point, we have to take into account that the MDS is also a beta implementation. Therefore, in a production environment this time could be lower.

7.4 Token-based Web Authentication

The components of the testbed described in this chapter were deployed to validate the authentication for web resources using the eduToken. The following features are demonstrated:

- Retrieving the encrypted eduToken from the secure store.
- Parsing the SAML statement within the eduToken.
- Displaying the token's content to the end-user in a GUI.
- Validating the eduToken according to the SAML XML schema and checking it's signature by the client.
- Providing an interface using Java's Remote Method Invocation (RMI) mechanism with a method to request the eduToken.
- Requesting the eduToken using a Java Applet using the RMI interface and POSTing it to the token servlet.
- Validating the eduToken by the token servlet.
- Creating a valid and signed Shibboleth Response.
- Sending the Shibboleth Response to the Shibboleth Service Provider.

- Use of certificates issued by the eduGAIN PKI.

Figure 7.7 shows all elements that are deployed for demonstrating the web authentication using the eduToken:

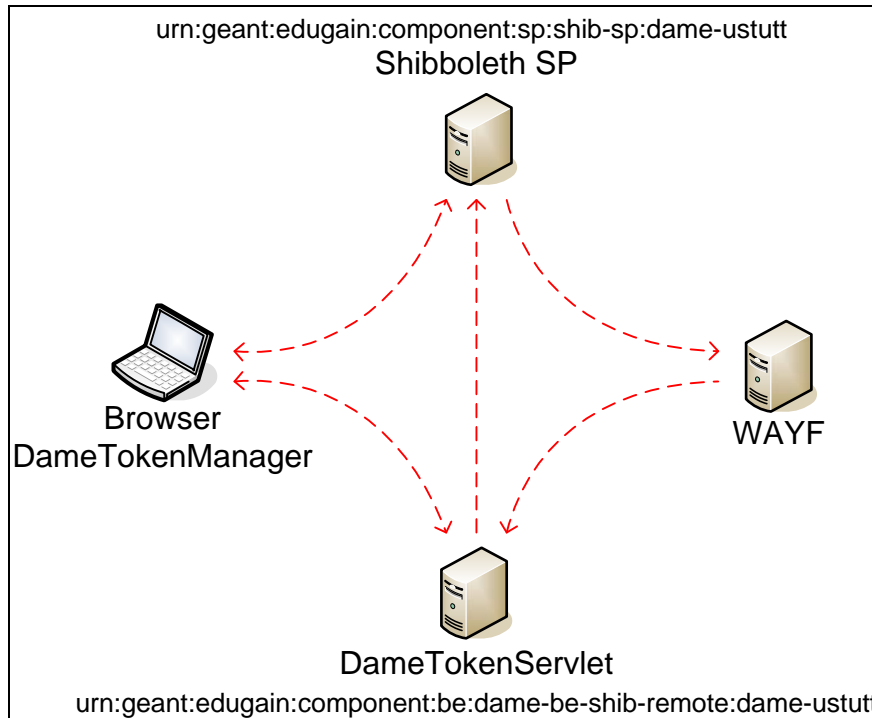


Figure 7.7: Testbed unified SSO part

The token manager, Shibboleth Service Provider (SP) and token servlet are deployed at University of Stuttgart, they are connected via a WAYF (Which Federation Are You From) service of the DFN to the eduGAIN testbed.

The elements are:

- DameTokenManager: The client part of the uSSO middleware. This links the network authentication with the web authentication, and provides a graphical interface to the end-user. The implementation is based on components and libraries of eduGAIN, in particular the OpenSAML library and the eduGAIN Validator. For the decryption operation the OpenSSL library is used. The Java code is compliant to JDK 5.0 to run on any Java Virtual Machine supporting this, irrespective of the operating system used.
- Shibboleth SP: A web server protecting a web resource and enforcing authentication via Shibboleth.
- WAYF: The SPs redirect user that have not yet been authenticated to this component so they can choose which IdP they want to use for authenticating, in this scenario the DameTokenFetcher and DameTokenServlet take the role of an IdP.
- DameTokenFetcher: The redirect of the WAYF goes to a website containing the DameTokenFetcher applet, which fetches the eduToken from the DameTokenManager and uses HTTP POST via an HTML form to send it to the DameTokenServlet.
- DameTokenServlet: This is a modified Shibboleth remote Bridging Element (r-BE) from the eduGAIN infrastructure that accepts eduToken from the DameTokenFetcher and then acting as an IdP sends a Shibboleth Response to the SP.

Detailed information about the hardware and software used to deploy this part is shown in the following table:

Element	Hardware	Operating System	Base Software	Extra libraries
DameTokenManager	Pentium T2400 2 GHz 1024 MB RAM	Windows XP Service Pack 2	Mozilla Firefox 2.0 Sun Java Runtime 1.5.0_13	eduGAIN 0.7 (with openSAML 1.2 and openSAML 2) not-yet-commons- ssl-0.3.8
Shibboleth SP	Pentium IV 3 GHz 1024 MB RAM	Windows 2003 SE Service Pack 2	Apache HTTP Server 2.0.61 Shibboleth SP 1.3f	
DameTokenServlet DameTokenFetcher	Pentium IV 3 GHz 1024 MB RAM	Windows 2003 SE Service Pack 2	Tomcat 5.5.25	eduGAIN 0.6 (with openSAML 1.2 and openSAML 2)

Table 4: Hardware and Software for Web Authentication

A number of tests have been performed successfully that demonstrate the functionality of the current implementation:

1. Accessing the protected web resource after deleting all browser cookies enforces Shibboleth authentication.
2. If the eduToken is available, the DameTokenManager is running, and the web resource is accessed, a webpage containing the DameTokenFetcher Applet is shown.
3. Pressing the button on that page the eduToken is shown in the Applet's Java console.
4. The eduToken is accepted by the DameTokenServlet and a Shibboleth Response is created.
5. The Shibboleth Response is accepted by the SP, access to the protected resource is granted.
6. Further accesses to the web resource are allowed without any interaction; the Shibboleth security context is established.
7. If one of the following applies, access to the resource is not granted: the eduToken is not available, the DameTokenManager is not running, the eduToken is not valid.

Precise time measurements of the delay possibly caused by the new token-based profile were not taken. No noticeably big extra delay was noticed during the test that would require optimisation for the components to become usable. Also the DAME components are still of prototype quality, and the used underlying eduGAIN infrastructure is still an alpha-release and under development at the time of writing, in contrast to the eduroam infrastructure.

7.5 Unified SSO and Authorisation for Grid

This section explains the components that were adapted and deployed for validating the Grid scenario of DAME; that is authentication via the eduToken to the Grid Portal and following authorisation for the Grid Service using the DAME PDP. Thereby it is shown that the DAME infrastructure can be used for the application-level service "Grid".

In order to test with a “real-life” grid service and portal, this has been developed in cooperation with the current EU-research Grid project ViroLab [ViroLab]. This project, co-participated by the University of Stuttgart, is developing grid services with the aim of creating a virtual laboratory for decision support in viral disease treatment. The ViroLab Grid Service is called Data Access Service. It allows distributed and heterogeneous data resources appear transparent to the virtual laboratory users by hiding them and their internals behind a layer of virtualisation services that provide access in a consistent, data resource-independent, and efficient way.

These features can be demonstrated with the deployed components:

- Authentication to the ViroLab GridSphere Portal, via the uSSO components of DAME using an available eduToken
- Performing authorisation when accessing the ViroLab Data Service by querying the DAME PDP and applying the installed Grid access policies

Again, no time measurements were taken as also the ViroLab software is still under development.

The required components that have been modified and deployed are:

- **GridSphere Portal & Shibboleth SP:** The GridSphere Portal is the resource protected behind the Shibboleth SP. As GridSphere runs on Apache Tomcat, the mod_jk connector is used for linking it to the Apache HTTP server and the Shibboleth SP, both running on the same machine. The GridSphere portal behaves from the Service Provider point of view as another resource in the Service Provider’s domain.
- In order to link DAME components to GridSphere, code has been added to the GridSphere software:
 - New “DAME Login” button on the login web page of the Portal
 - New DAME portlet, enabling authentication through the Service Provider and obtaining the user’s handle and the IdP (for later authorisation). It consists of JSP files, Java code and XML configuration data. It functions as a GT4 client.
- **Grid Data Service:** The grid data service is an implemented OGSA Service that is built with the help of the Globus Toolkit framework. The Grid Data Service uses the OGSA-DAI middleware for accessing different types of Data Access Points. The Grid Data Service has been extended to perform an authorisation query to the DAME PDP. At this PDP policies for controlling access to the Grid resources are deployed. These policies are applied on the user’s attributes, received from the DAME portlet, to result in an authorisation decision.

Detailed information about the hardware and software used to deploy this part is shown in the following table:

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

Element	Hardware	Operating System	Base Software	Extra libraries
GridSphere	Pentium IV 3 GHz 1024 MB RAM	Windows XP Service Pack 2	Tomcat 5.5 Sun Java Runtime 1.5.0_13	Modified to be able to accept token based web profile
GridPortlet	Pentium IV 3 GHz 1024 MB RAM	Windows XP Service Pack 2		GWT
Shibboleth SP	Pentium IV 3 GHz 1024 MB RAM	Windows XP Service Pack 2	Apache HTTP Server 2.0.61 Shibboleth SP 1.3f	mod_jk
Grid Data Service	Pentium IV 3 GHz 1024 MB RAM	Windows XP Service Pack 2	GT4 WS-Core Python JSE 1.5	OpenSAML WSRF libraries OGSA-DAI
PDP	Pentium IV 3 GHz 1024 MB RAM	Ubuntu Linux Kernel 2.6.15	Tomcat 5.5.12	openSAML 1.1. sunXACML 1.2

Table 5: Hardware and Software for Grid

For the following authorisation use cases XACML policies have been developed and the corresponding tests were successfully executed:

1. Doctor working at a Hospital as Staff
2. Virologist working at a Research Centre as Staff
3. Researcher working at a University as Staff

The attributes corresponding to these use cases are *urn:mace:dir:attribute-def:virolabRole* with the values "Doctor", "Virologist" and "Researcher", *urn:mace:dir:attribute-def:affiliation* with the value "Staff" and *urn:mace:dir:attribute-def:homeOrganizationType* with the values "Hospital", "Research Centre" and "University". These attributes and values are of course specific and meaningful to the ViroLab Data Service.

An unauthorised user can only list all databases, but not access any. In cases 1 and 3 access to two Databases A and B is given, in case 2 access to a Database C is given. The authorised users can list the database structures and contents, and submit queries that can be distributed ones (querying several of the accessible databases at different locations). An example distributed query is *select * from nucleotide_sequences*. This statement would select all columns and rows of the table *nucleotide_sequences* of all accessible databases.

8 Conclusions

The main goal of the DAME project was to show how a real and widely-deployed user federation for an inter-NREN network roaming service, such as eduroam, can be used as an initial element to bootstrap a unified Single Sign On architecture. This requires the extension of the standard eduroam with eduGAIN-based elements and the design of specific profiles for different application scenarios, ranging from network access control to Grid computing. As is shown in this document, the DAME project has provided a particular design, implementation and validation of a uSSO architecture. It has been demonstrated that eduGAIN constitutes an excellent mechanism to incorporate SSO mechanisms at a confederation level and for different high level scenarios. Furthermore, the current proposal puts in common several federation initiatives, such as eduroam, Shibboleth and PAPI, in order to build the final prototype. Time measurements have been presented showing that the overhead introduced by the authorisation process and the token delivery is not substantial, therefore obtaining an acceptable response-time for scenarios like network authorisation.

In relation to future work derived from the uSSO architecture, we should take into account the support for different levels of security required to access protected resources. Several organisations are interested in the definition of the authentication strength required to be assured that an entity is indeed the claimed entity, which is known as *Level of Assurance (LoA)*. Taking advantage of the points of extension provided by the eduGAIN infrastructure, a study of the necessary elements, policies and messages to enable the use of LoAs in the current uSSO architecture should be done.

9 References

- [8021X]** LAN MAN Standards Committee of the IEEE Computer Society. Standard for Port based Network Access Control. IEEE Draft P802.1X/D11. March 2001.
- [AAA]** C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, D. Spence. Generic AAA Architecture. Request for Comments (RFC) 2903. August 2000.
- [AAICOOK]** Best Practice Guide - AAI Cookbook -First Edition Guidelines for Connecting to the eduGAIN AA Infrastructure
- [AUPOL]** Australian eduroam Policy Version 2.0. April 2 2007.
- [DIAM]** P. Calhoun, J. Loughney, E. Guttman, G. Zorn, J. Arkko. Diameter Base Protocol, September 2003. Request for Comments (RFC) 3588.
- [EAP]** B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowitz. Extensible Authentication Protocol (EAP). Request for Comments (RFC) 3748. June 2004.
- [EDUPER]** EduPerson Specification (200312) <http://www.educause.edu/eduperson/>
- [EDUSERV]** M. Milinovic, Juergen Rauschenbach, Stefan Winter, Licia Florio, David Simonsen, Josh Howlet. Deliverable DS5.1.1: eduroam Service Definition and Implementation Plan. January 2008.
- [EG]** Best Practice Guide - AAI Cookbook -First Edition Guidelines for Connecting to the eduGAIN AA Infrastructure
- [ER]** eduroam website. <http://www.eduroam.org>
- [GLOSS]** Manuela Stanica, Torbjörn Wiberg, Klaas Wierenga, Stefan Winter, Juergen Rauschenbach. Milestone MJ5.1.5: JRA5 Glossary of Terms - Second Edition - update of DJ5.1.1. January 2007.
- [GS]** GridSphere Project website: <http://www.gridisphere.org/gridsphere/gridsphere>
- [GT]** Globus Toolkit Homepage: <http://www.globus.org/toolkit/>
- [NAS]** G. López, O. Cánovas, A.F. Gómez, J.D. Jimenez, and R. Marín. A network access control approach based on the AAA architecture and authorization attributes. Journal of Network and Computer Applications JNCA, 2006.
- [OGSA]** Globus Open Grid Services Architecture: <http://www.globus.org/ogsa/>
- [PAPI]** R. Castro-Rojo, D.R. López. The PAPI System. Point of Access to Providers of Information. TERENA Networking Conference. 2001.
- [PEAP]** A. Palekar, D. Simon, J. Salowey, H. Zhou, G. Zorn, S. Josefsson. Protected EAP Protocol (PEAP) Version 2. Internet draft. October 2004.
- [POLI]** López, G., Cánovas, O. and Gómez-Skarmeta, A.F. Use of XACML policies for a Network Access Control Service. Proceedings of 4th International Workshop for Applied PKI (IWAP 05), pp. 111-122. 2005.

- [RAD]** C. Rigney, S. Willens, A Rubens, W. Simpson. Remote Authentication Dial In User Service (RADIUS). IRTF RFC 2865. June 2000.
- [SAML]** OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1. OASIS Standard. September 2003.
- [SCHAC]** SCHAC Schema Harmonisation Committee Attribute Definitions For Individual Data Version 1.2.0 4 May 2006
- [SHIB]** T. Scavo, S. Cantor. Shibboleth Architecture. Technical Overview. Working Draft 02. June 2005.
- [TOKEN]** David Lutz. Secure AAA by means of Identity Tokens in Next Generation Mobile Environments. In: Proc. of the Third International Conference on Wireless and Mobile Communications (ICWMC), 2007
- [USSOREQ]** B. Kerver, M. Stanica, J. Rauschenbach, K. Wierenga. Deliverable DJ5.3.1: Documentation on GÉANT2 unified Single Sign-On (uSSO) Requirements. February 2007.
- [Virolab]** ViroLab website: <http://www.virolab.org/>
- [VOMS]** R. Alfieri et al. VOMS, an Authorization System for Virtual Organizations. In Proc. of the 1st European Across Grids Conference, Santiago de Compostela, 2003.
- [WSRF]** OASIS Web Services Resource Framework website: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf
- [XACML]** OASIS. eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard. February 2005
- [XSUPPLICANT]** Official web site: <http://open1x.sourceforge.net/>

10 Acronyms

AAA	Authentication, Authorisation and Accounting
AAI	Authentication and Authorisation Infrastructure
AP	Access Point
AS	Authentication Server
BE	Bridging Element
DAMe	Deploying Authorisation Mechanisms for Federated Services in the eduroam Architecture
EAP	Extensible Authentication Protocol
eduroam	Education Roaming
HTTP	Hypertext Transfer Protocol
IdP	Identity Provider
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
MDS	Meta Data Service
NAC	Network Access Control
NREN	National Research and Education Network
PDP	Policy Decision Point
PEAP	Protected Extensible Authentication Protocol
PoA	Point of Access
QoS	Quality of Service
RADIUS	Remote Authentication Dial In User Service
SAML	Security Assertion Markup Language
SCHAC	Schema Harmonisation Committee
SP	Service Provider
SSID	Service Set Identifier
SSO	Single Sign On
TERENA	Trans-European Research and Education Networking Association
TLS	Transport Layer Security
TLV	Type Length Value
TTLS	Tunnelled Transport Layer Security
uSSO	Unified Single Sign On
VPN	Virtual Private Network
WAYF	Where Are You From
WFAYF	Which Federation Are You From
XACML	eXtensible Access Control Markup Language

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2

Project:	GN2
Deliverable Number:	DJ5.3.2
Date of Issue:	02/06/08
EC Contract No.:	511082
Document Code:	GN2-08-080v2