



Authentication and Authorization Infrastructure (AAI)

## **Architecture Evaluation**

## Document management

Version/status: 1.0

Date: 20-Jan-03

Author(s):	Serge Droz	PSI
	Rolf Gartmann	SWITCH
	Christoph Graf	SWITCH
	Daniela Isch	at rete ag
	Claude Lecommandeur	EPFL
	Gerhard Hassenstein	UAS Bern
	Thomas Lenggenhager	SWITCH
	André Redard	at rete ag

File name: AAI\_Arch\_Eval\_v10.doc

Replacing: 0.5 / 11-Jan 03

Approved by:

## Table of Content

<b>1.</b>	<b>Introduction</b>	<b>4</b>
<b>2.</b>	<b>Architectures</b>	<b>5</b>
2.1	Shibboleth	5
2.2	PAPI	7
2.3	Tequila	9
<b>3.</b>	<b>Evaluation Methodology</b>	<b>11</b>
<b>4.</b>	<b>Evaluation Results</b>	<b>11</b>
4.1	Shibboleth 0.7	12
4.2	PAPI 1.2	12
4.3	Tequila 1.1	13
4.4	Long-term Issues	13
4.5	Conclusion	13
	<b>Appendix A Evaluation Sheet</b>	<b>14</b>

## Figures

Figure 1: Shibboleth interactions	6
Figure 2: PAPI interactions	7
Figure 3: Tequila interactions	9

## Tables

Table 1 Evaluation results	11
----------------------------	----

## 1. Introduction

In September 2001, an inter-university study group published the “AAI Concept”, a report proposing a roadmap to develop and implement an Authentication and Authorization Infrastructure (AAI) for the higher education community in Switzerland. SWITCH, in collaboration with specialists from this community, subsequently took on the task to implement the phase “preparatory study” as outlined in the concept. Between December 2001 and May 2002, several architectures were put to close scrutiny with regard to technical, organizational, financial and legal aspects. It turned out that there were feasible solutions available, so that it was decided to further investigate the two most promising architectures, Shibboleth und PAPI.<sup>1</sup>

A task force was set up to evaluate the architectures. Since it soon appeared that PAPI would cause considerable problems with respect to attribute transfer, most parties decided to start their pilot projects with Shibboleth rather than with PAPI. Unfortunately, the developers of Shibboleth repeatedly did not meet the deadlines they had set themselves. This delayed the start of the pilot projects considerably so that in the end, there was not as much data or practical experience available for the evaluation as expected.

The architecture GASPARG had been ruled out in the preparatory study, because it was designed for intra-organizational use only. In the meantime, however, the École Polytechnique Fédérale de Lausanne (EPFL) had developed GASPARG further to make it suitable for inter-organizational use. After Tequila, as this new architecture was named, was examined and proved to be in line with the technical and legal requirements, the project management decided that Tequila was also to be considered in the evaluation.

The following chapters describe the three architecture candidates, show the evaluation methodology as well as the evaluation results and give a recommendation.

---

<sup>1</sup> For details see the report on the Preparatory Study at [www.switch.ch/aaai](http://www.switch.ch/aaai)

## 2. Architectures

### 2.1 Shibboleth

Shibboleth<sup>2</sup> is a joint project of Internet2/MACE (Middleware Architecture Committee for Education)<sup>3</sup> and IBM. It aims to develop an architecture for standard-based vendor-independent web access control infrastructure that can operate across institutional boundaries.

The requirements on which Shibboleth was designed are documented in <http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-requirements-01.html>

#### 2.1.1 Architecture / System Design

The primary design principles for Shibboleth are:

- No single central piece of infrastructure required, scalable
- Data protection and privacy are of importance for Shibboleth
- The user is guided by 'HTTP redirect' from the resource to the authentication server and back to the resource for the authorization

A detailed description of the Shibboleth architecture can be found in *Shibboleth-Architecture Draft v0.5*<sup>4</sup>.

Shibboleth uses a federated administration; a Resource Owner leaves the administration of user identities and attributes to the user's Home Organization, which is also responsible for providing attributes about a user (possibly but not necessarily including a username) that the Resource Owner can use in making an access control decision when the user attempts to use a resource. Users are registered only at their Home Organizations, and not at each resource.

Shibboleth is a system for securely transferring attributes about a user from the User's Home Organization to the site of the Resource Owner, provided the resources are accessible via standard web browsers. In addition, Shibboleth enables the users to decide which information about them gets released to which site. The users therefore have to balance access and privacy.

The major components of Shibboleth are:

---

<i>WAYF</i>	Where Are You From Server
	Redirects the user back to the HS at his/her Home Organization. At least one WAYF server is needed, but it may be replicated as desired.
<i>HS</i>	Handle Server
	Authenticates a local user according to the methods of the Home Organization and provides an opaque handle identifying the user.
<i>AA</i>	Attribute Authority
	Retrieves the attributes which a user allows to be given to a resource (according to the user's Attribute Release Policy) and passes them to the SHAR on behalf of the resource.

---

<sup>2</sup> <http://shibboleth.internet2.edu>

<sup>3</sup> <http://middleware.internet2.edu/MACE/>

<sup>4</sup> Shibboleth-Architecture DRAFT v05, Marlena Erdos and Scott Cantor, May 2, 2002  
<http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-v05.pdf>

<i>SHIRE</i>	Shibboleth Indexical Reference Establisher  Makes sure that the resource gets a 'pointer' (handle) back to the user without requiring more knowledge about a user. In case it is missing it refers the user via the WAYF server back to his/her HS to get one.
<i>SHAR</i>	Shibboleth Attribute Requester  Contacts the AA to fetch the available attributes describing the user and passes them on to RM.
<i>RM</i>	Resource Manager  Decides on access to the resource based on the information received and where necessary the information about earlier sessions of the same user.

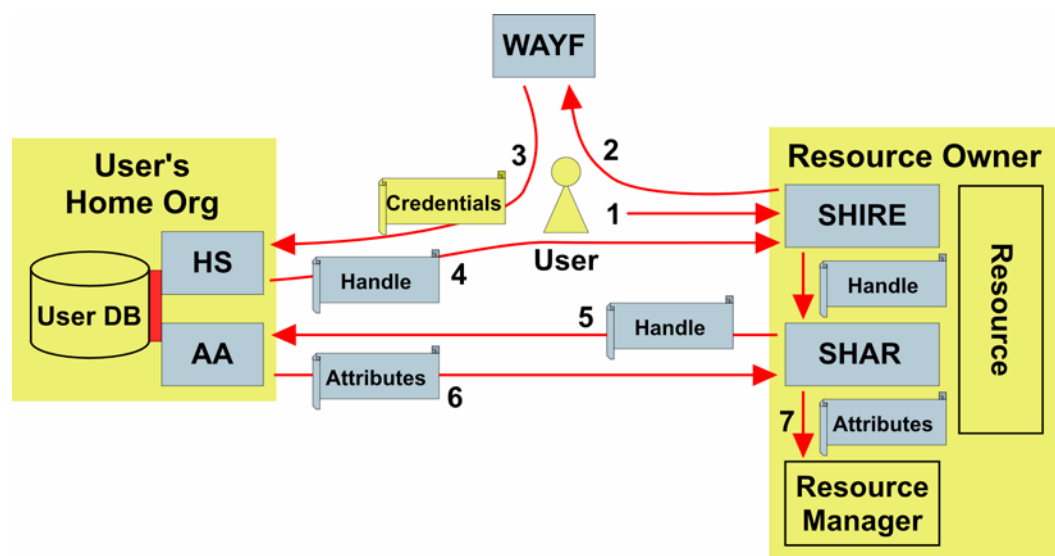


Figure 1: Shibboleth interactions

Example of Shibboleth usage:

A user U, affiliated to the Home Organization O, wants to access a web-based resource R located at some remote site.

- U connects with his or her web browser to the web site R (1). The server R does not detect the required authorization information and redirects U (2) to the 'Where Are You From' web server W. The URL of R gets passed along.
- On W, U selects his/her Home Organization O from a list of organizations participating in Shibboleth. W redirects U (3) to the web server HS (Handle Service) located at the Home Organization O. The URL of R gets passed along again.
- U authenticates him-/herself according the local rules and methods towards HS. Once authenticated, HS generates an opaque handle H for the user U. H is the authentication info U needs to present to R. U gets redirected to R (4).  
R sends handle H together with the URL of R (5) to the Attribute Authority (AA) located at the Home Organization O.  
AA checks which Attribute Release Policy (ARP) of user U applies to resource R. AA returns the attributes it is allowed to send to R (6).

Within R the attributes retrieved get passed to the Resource Manager RM (7) that decides on providing access.

- U gains access to the resource

### 2.1.2 Current Status

After the rewriting of the Shibboleth components at the resource side, the first official release v0.7 has been made available end of November 2002. The current schedule is to release v0.8 by March 1, and v1.0 before the Internet 2 meeting (April 19, 2003). In v0.8 the major functional improvements will happen at the Attribute Authority and the way Attribute Release Policies are managed

## 2.2 PAPI

PAPI is a system for providing access control to restricted web-based information resources across the Internet. It intends to keep authentication as an issue local to the User's Home Organization, 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 organization to use its own authentication schema, maintaining user privacy, and offering information providers the attributes required for access control decisions. Moreover, access control mechanisms are transparent to the user and compatible with the most commonly employed Web browsers, i.e., Netscape/MSIE/Lynx, and any operating system.

PAPI has been designed and is being developed by a small team from the Spanish national research network RedIRIS. Descriptions and the product itself can be found at this location: <http://www.rediris.es/app/papi/index.en.html>

### 2.2.1 System Architecture

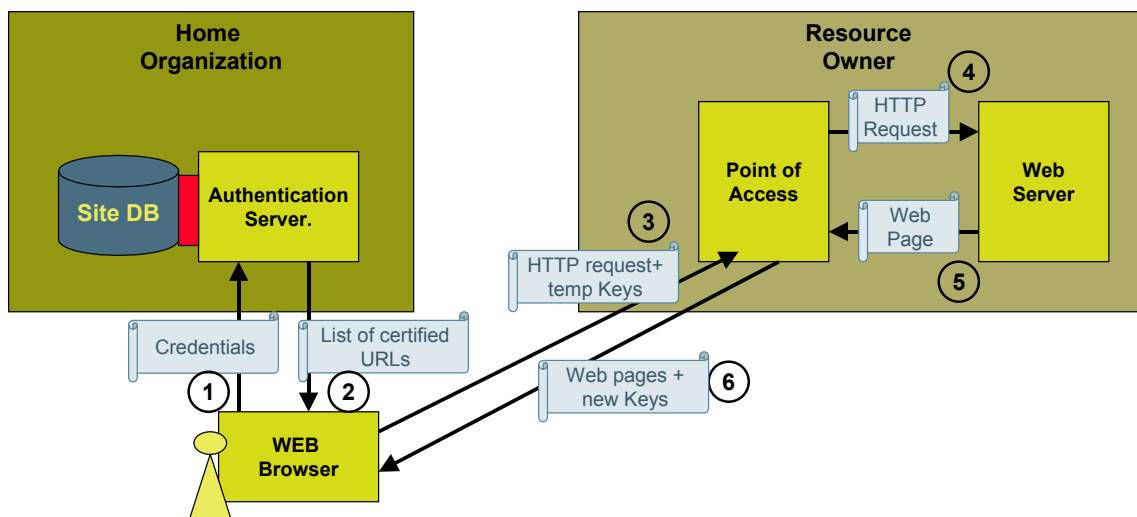


Figure 2: PAPI interactions

System components (see Figure 2)

- Authentication server (AS): The user has to provide credentials to the AS which are in turn verified against the organization's authentication schema (e.g. LDAP, POP, x.509-certs, etc.) [1]. Once successfully authenticated, the AS will consult a local site database and create a web page listing all web resources available to the user. Cryptographically signed authorization information is also coded into those URLs [2].
- User's web browser: While building up the page received from the AS, the user's browser will contact all points of access of those web resources and thus provide them with authorization information.[3]
- Point of access (PoA): This component performs actual access control to a set of protected web resources. It verifies the authorization information it receives from the user's browser and updates cookies in the user's browser to keep track of authorized users [6]. The PoA acts as a dedicated web gateway as shown above and accesses the resource on behalf of the user's browser [4][5]. Alternatively, the PoA can be integrated with the resource as an access control module of the resource's web server.
- Protected web resource: It trusts its PoA(s) to perform access control on its behalf.
- Version 1.1 of PAPI introduced the concept of a GPoA (Group wide PoA, not shown in Figure 2). A GPoA acts as gateway to a set of PoAs with identical access policy. The user gets access to all resources, if the browser initially exchanged keys with the GPoA in charge of those resources. This is an important feature to increase scalability of the PAPI model. Resource hosting organizations are advised to define a short list of different access policies and group all their resources behind a low number of GPoAs, one for each access policy.

Since the AS has to know about all resources available to the user, we propose to set up a central database listing all PAPI-enabled resources and their access policies. This will help AS operators to compile a list of resources available to their users.

#### Example of PAPI usage

A user, affiliated with Home Organization, wants to access a web-based resource located at some remote site.

- 1) The user authenticates him-/herself to the AS of the User's Home Organization with whatever method the Home Organization uses for that purpose. The user gets a list of available resources.
- 2) While the page is being built up, the browser contacts all PoAs of all resources with attribute information embedded in the URL. Each point of access evaluates the received attributes and checks them against a local access policy. Graphical elements in the user's browser window will inform about the outcome of that authorization check for each entry in the list of available resources. The PoA will at the same time send cookies to the user's browser allowing the user to access the resource without renewed authentication.
- 3) The user can now access all listed resources without any further authentication and will only be redirected back to the authentication server after expiry of the cookies.

### 2.2.2 Current status

With the current version 1.2, released on November 15, 2002, some features regarding the privacy aspect have been added. The AS can now be configured to send individual assertions about a user to each (G)PoA. They are built based on the configuration attributes 'papiQualifiedAssertion' and/or 'papiAssertion' and can be defined user- and resource-specific.

Therefore, not all attributes are transmitted to all resources anymore. Filters at the PoA side can now be configured to reject or allow a request based on the received attributes, contrary to only reject a request as was the case in version 1.1.

The PAPI authors intend to integrate into a future version of PAPI the option that the AS issues lists of resources without instant attribute exchange with all PoAs. The attribute exchange will only take place for the resource the user wants to access. This will further improve scalability and eliminate a potential data protection issue.

### 2.3 Tequila

GASPAR, the AAI of EPFL, has been developed for and is in use at a single organization. It offers not only AAI functionality but also user directory and authentication.

In November 2002, EPFL has started to write a modular AAI based on GASPAR. The new system is called "Tequila" ("T'es qui là?") and is designed for a federated environment.

Tequila is available as version 1.1 from the developers at EPFL. It is well documented and has been installed in a test lab at SWITCH, but hasn't been tested in a pilot environment yet.

#### 2.3.1 Architecture / System Design

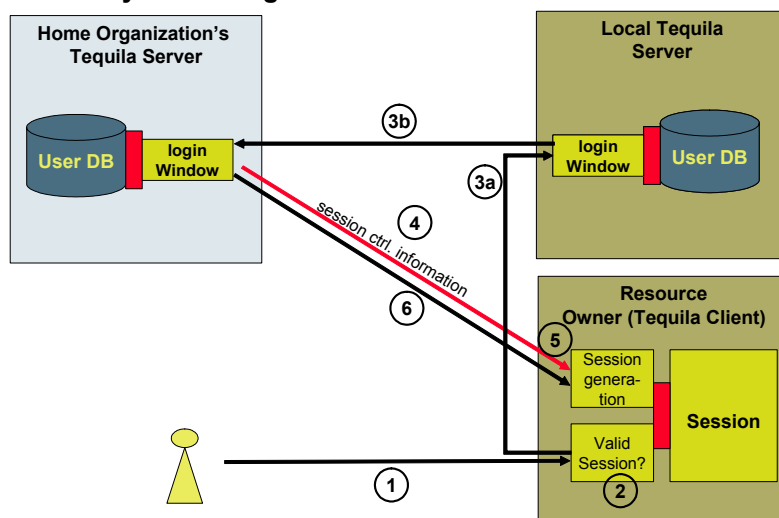


Figure 3: Tequila interactions

Tequila consists of the Tequila server(s) for authentication at the Home Organization side and the Tequila client(s) for authorization at the resource location. The authentication mechanisms are designed to be as flexible as possible. Required attributes to allow access to a resource are requested from the Tequila client side.

**Authentication:**

Before a user accesses a Tequila-protected resource, he/she must register with the Home Organization. To authenticate the registered users, the Tequila server offers interfaces towards LDAP and SQL; additional interfaces should be easy to integrate.

**Resource Access:**

(1) A user tries to access a web-resource.

(2) The resource checks whether the user has a valid session before granting access to protected data. If the session does not exist or has expired, the resource uses Tequila's open API to authenticate the user.

(3a) The resource redirects the request to the local Tequila server's login window and provides necessary parameters (including the choice to switch to the user's Home Organization Tequila server (3b), implementing a 'Where Are You From' functionality). The Home Organization Tequila server authenticates the user by login and password, a digital certificate, or automatically by cookie, if present. This is done by using the existing authentication service of the Home Organization.

(4) If the user is authenticated, Tequila sends the authorization attributes and a session identifier to the resource server. This out-of-band interaction is done via a HTTP get or put, i.e. without involving the user's browser.

(5) Upon reception of authentication attributes, the resource server creates a new session for the user (based on the session ID from Tequila and with the user's identification and a timestamp).

(6) Tequila then redirects the user's browser to the resource server with session ID as a parameter. The resource server checks the session and grants access to protected services.

It is then up to the resource to ensure the security of the communication and a proper access control (identity, session ID, expiration of the session).

### 3. Evaluation Methodology

As mentioned in the introduction, the start of the pilot projects was delayed so that they could not provide as much input as expected. Thus the evaluation was mainly based on experience with the test installations of Shibboleth 0.7, Tequila 1.1 and PAPI 1.1 run by SWITCH as well as on the available documentation. SWITCH's feedback to the developers of PAPI regarding problems with the attribute transfer prompted them to come up with a revised version 1.2; while the test installation was not changed thereupon, the documentation that was consulted for the evaluation was the new one.

As a first step in the evaluation process, the criteria in the categories *functionality*, *ease of integration*, *architecture/security*, and *availability/support* were defined and weighted. The categories themselves were weighted as well.

The entire evaluation sheet is attached as Appendix A .

### 4. Evaluation Results

The formal evaluation as defined in chap. 3 lead to the following results (see Appendix A for further details):

Category	Type of criteria	PAPI	Shibboleth	Tequila
Functionality	Minimal requirements	Fulfilled	Fulfilled	Fulfilled
Ease of Integration	Minimal requirements	Not fulfilled (IIS Support)	Not fulfilled (IIS Support)	Not fulfilled (IIS Support)
	Weighted criteria	7.4	8.0	7.1
Architecture / Security	Weighted criteria	5.9	7.7	8.2
Availability / Support	Weighted criteria	6.4	5.4	4.7
<b>Overall rating</b>		<b>6.6</b>	<b>7.0</b>	<b>6.7</b>

Table 1 Evaluation results

The evaluation didn't show a clear leader; and, compared with an ideal solution, all three architectures have their shortcomings.

There turned out to be one mandatory criteria which was not fulfilled by either architecture, namely that the AAI has to provide native support<sup>5</sup> for IIS (>=Rel 4). Nevertheless, this non-fulfillment was not deemed a show stopper, because there are either ways to get round the problem (for PAPI) or solutions seem to come up (for Shibboleth and Tequila).

The following chapters list the most important pro's and con's of each architectures as well as various long-term issues (2-3 years from now).

---

<sup>5</sup> "Native support" means that a configurable, documented plug-in exists.

## 4.1 Shibboleth 0.7

### Pro's

- Wide support, because it is an Internet 2 project
- International interest and good awareness already
- Commercial Resource Providers are interested in integration of Shibboleth technology
- Wide basis of test users owing to international interest; therefore better detection and elimination of bugs
- Good integration into Apache environment
- Based on eduPerson LDAP schema
- Influence at Liberty Alliance by 'invited experts'
- Powerful and standardized protocol for transport of authentication and authorization information (SAML)

### Con's

- In pilot phase only
- Not all features planned in the design yet implemented, especially attribute release policy still rudimentary
- Attribute authority has to be modified for Swiss attributes (swissEduPerson)
- Deadlines for releases not met so far
- Support organization not clear
- Difficult to install because of strong dependency on platform (Apache / OS Release)

## 4.2 PAPI 1.2

### Pro's

- Only architecture that is already in operation (yet mainly within one organization and mostly for library resources)
- Good integration in Apache environment
- Good proxy support for "black boxes"
- External support being planned

### Con's

- User has no control of attribute release policy
- Home Organization has to know Resources
- Use of mod\_perl necessary at Resource (maintenance problematic)
- Certificate management requires more work (every Resource needs certificates of Home Organizations)
- Special LDAP schema necessary
- Perl approach (configuration susceptible to mistakes; awkward debugging )

### 4.3 Tequila 1.1

#### Pro's

- Lean, flexible solution
- Quick changes possible, because it is a flexible local development
- User has full control of attribute release policy
- Flexible LDAP schema usable
- Predecessor GASPARD in operation at EPFL

#### Con's

- No / very small awareness in international community
- No test users basis (except for GASPARD)
- No adaptations by commercial Resource Providers to be expected
- Long-term development unclear, no roadmap
- Commitment of EPFL for further development and support missing so far (budget?)
- Perl approach (configuration susceptible to mistakes)
- Missing examination with regard to security vulnerabilities (transference of experiences with GASPARD to Tequila is only possible with reservations, because design and code have been changed considerably)

### 4.4 Long-term Issues

- At present, Shibboleth does not yet offer the entire planned functionality; however, it is seen as some sort of "reference architecture".
- The future prospects of a widely supported architecture (primarily Shibboleth, but also PAPI) are better than those of a local approach (Tequila); yet the possibility to influence the development of the latter is better.
- Future changes in an architecture are better supported if it has a large user community, which is the case for Shibboleth. For such architectures, update scenarios can be expected.
- PAPI intends to approach Shibboleth and become compatible, but has a different philosophy than Shibboleth and Tequila as regards the Home Organization's side; possible interim scenarios are not clear today.
- From what is known so far, Tequila appears to meet today's requirements. Yet doubts exist as to long-term international compatibility; for while content providers and software vendors will most probably offer complete standard solutions for Shibboleth, there will be no such support for Tequila.
- Because of similarity in architectures, a migration from Tequila to Shibboleth would be easier than from PAPI to Shibboleth (at least as far as the Home Organization's side is concerned).

### 4.5 Conclusion

PAPI doesn't offer any significant advantages today and bears the risk of a costly migration in the future.

Choosing Shibboleth means that one has to make concessions regarding attributes able to be released to resources at a first stage, yet with the advantage of hopefully having an internationally supported architecture.

If these concessions are not acceptable, Tequila is the preferable choice at present, yet at the risk of having to migrate to an internationally supported architecture (probably Shibboleth) later on, which will cause additional expenditures/costs. It goes without saying that a formal commitment of EPFL to support and enhance Tequila would be required.

## Appendix A Evaluation Sheet

		Must,Wish	Importance	Weight	PAPI ( Version 1.2 )		SHIBBOLETH ( Version 0.7 )		TEQUILA ( Version 1.1 )		
Functionality		M/W	1-10	%	Description	Yes/No; 1-10	Description	Yes/No; 1-10	Description	Yes/No; 1-10	
	Functional model	AAI has to provide the core functionality defined in the AAI model.									
1.01		Home Organization: authenticates users and maintains user directory, containing user attributes	M			Yes		Yes		Yes	
1.02		Authorization attributes: transferred from Home Org to Resource by AAI	M			Yes		Yes		Yes	
1.03		Resource Owner: controls access to resource	M			Yes		Yes		Yes	
1.04		AAI Core functionality: coupling user authentication, resource access and transport of authorization attributes by means of secure methods	M			Yes		Yes		Yes	
<b>Ease of integration</b>											
2.01	OS @ Home Org	Minimal requirement: AAI @ Home org has to run at least on one of these platforms: Linux, Solaris, Win2000/XP	M			Yes	tested on Linux; Perl based, could run on W2K	Yes	tested on Linux, could run on W2K	Yes	
2.02	Interfaces @ Home Org	Minimal requirement: AAI @ Home Org has to provide at least interfaces to authentication system and to user directory based on LDAP or well defined API	M			Yes	interfaces to POP; IMAP; flat file; LDAP available	Yes	available interfaces dependend on available mod_auth Apache modules	Yes	LDAP Interface available
2.03	Interface @ resource	Minimal requirement: AAI @ Resource has to provide at least native support for IIS (>=Rel 4) and Apache (>=Rel 1.3.2), proxy support for "black box" web resources (native support means: configurable, documented plug-in)	M			No	mod_perl based; good proxy support; no native support for IIS at the moment; planned for Spring 03	No	IIS not yet, but to come; proxy support unknown	No	tested on Linux, could run on IIS, no proxy support
2.04	Ease of integration @ Home Org	Ease of Home Org integration: support for additional OS (any unix flavor) and common interfaces to authentication systems (e.g. radius) and user directories (e.g. sql)	W	5	11%	5	special LDAP schema necessary and available (papi_schema); auth towards POP; IMAP; flat file, LDAP available	6	Solaris, Linux. Authentication: LDAP, MySQL, any methode based on mod_auth apache modules (REMOTE_USER), interfaces for Authorization Attributes are available for LDAP / MySQL	4	LDAP, SQL Interface available, Hooks can be used for additional authN, authZ modules
2.05	Ease of integration @ Resource	Ease of resource integration: configurable (instead of programmable) integration with a variety of common web resources (IIS, Apache, "black box" web resources; personalized and unpersonalized resources) on unix flavors, Win2K/XP	W	9	20%	6	integrated through mod_perl; proxy_mod for 'black box' resources available; configuration based on Apache Config	4	Solaris, Linux. Apache support very good. IIS not yet supported. Others need to be programmed, ev. with help of exported environment variables.	4	Perl Module, Java Code, raw Interface available, configurable with mod_rewrite; coding at resource location mostly necessary
2.06	End user system	Runs on major end user platforms (Windows 9x - XP, MacOS 9X, unix flavors with any HTML 3.2 compatible browser with https and cookie support, e.g. IE, Mozilla, Opera, Lynx, Netscape); no need for additional software on end user system.	W	9	20%	7	cookie support heavily used; cookie from everywhere sometimes needed	9	Browsers need to support redirection, SSL and cookies.	9	cookies not needed
2.07	PKI	No need for end-user carried certificates	W	8	17%	10	no need	10	no need	10	no need
2.08	End-User Hardware	No need for enhanced end-user hardware (e.g. smart card reader)	W	10	22%	10	no need	10	no need	10	no need
2.09	International Communities	Support from large software vendors and international communities	W	5	11%	4	mainly in Spain established	8	WebCT, EBSCO part of Pilots; international awareness; kind of reference architecture	2	no international awareness
				46	100%	7.4		8.0		7.1	

			Must,Wish	Importance	Weight	PAPI ( Version 1.2 )		SHIBBOLETH ( Version 0.7 )		TEQUILA ( Version 1.1 )	
Architecture, Security			M/W	1-10	%	Description	Yes/No; 1-10	Description	Yes/No; 1-10	Description	Yes/No; 1-10
3.01	Privacy	Transparent attribute release policy: user should see, which attributes are transmitted to a specific resource (before transmission)	W	6	8%	user is not aware of release policy	1	Not implemented yet. Don't know if they plan to implement that in the near future. Don't see why this should not be possible technically.	5	users sees and can choose which attributes should be released to a resource	10
3.02	Privacy	AAI should provide a mechanism (filters) which allows to restrict the transmission of attributes to a minimal set (resource specific)	W	5	6%	possible in 1.2.0 with papiQualifiedAssertions	6	Provides a mechanism (filters) which allows to restrict the transmission of attributes to a minimal set (user, resource, site specific)	10	no filters at HomeOrg site implemented	5
3.03	Resource catalog	No need for maintaining a centralized Resource Catalog	W	7	9%	no, but Resources are still on the AS side to be defined	2	some mechanism needed to define Attribute Release Policies at HomeOrg	6	not needed	10
3.04	Decentralization	No need for an AAI-wide, centralized user directory	W	9	12%	not needed	10	not needed	10	not needed	10
3.05	NAT	AAI must be able to serve users behind NAT devices and Web Proxies.	W	9	12%	no problems	10	no problems	9	no problems	10
3.06	Firewalls	Ease of describing the AAI traffic patterns for integration into firewall rules (for firewalls between User, HomeOrg, ServiceProvider and Resource)	W	8	10%	no problem if resource has no fw issues	10	no problem if resource has no fw issues	9	no problems, if resource has no fw issues	10
3.07	Security	AAI must provide mechanism to guarantee Privacy, Message Integrity, Non-repudiation, Authentication of AAI systems. Transfer of private data must be encrypted (e.g. SSL)	W	10	13%	message integrity done with short & longterm encrypted cookies, verified at the resource, cookie stealing for short periods seems possible, SSL supported	7	Can't see any conceptual security problems.	10	repudiation possible if only http is used ( URL stealing )	6
3.08	Security Patches	Ease of applying security patches (OS, libraires, packages as well as to the AAI software itself)	W	7	9%	Perl, mod_perl	5	Solaris, Linux. Only standard technologies, but perhaps a bit many of them: Apache, CPP, Tomcat, JDK.	4	Perl	8
3.09	Scalability	AAI must scale regarding 100 Home Organizations, 300000 users and 10000 resources within Switzerland: no (# of HomeOrg X # of resources)-dependencies	W	8	10%	information about resources still hold at AS ( HomeOrg ); PoA need to have Certificates of all AS	2	Fully scalable. Ev some issues with the server certs.	8	architecture seems scalable	8
3.10	Performance	No negative impact on performance of user interaction with resource (similar user experience with and without AAI)	W	6	8%	hard to tell	5	a lot of steps involved but hard to tell how the impact is	7	hard to tell; mod_rewrite involved	7
3.11	Application-Application AAI	AAI should provide mechanism to authenticate/authorize applications (instead of users), e.g. for accessing Web Services, WebDAV	W	3	4%	unknown at the moment	1	unknown at the moment	1	unknown at the moment	1
				78	100%		5.9		7.7		8.2

			Must,Wish	Importance	Weight	PAPI ( Version 1.2 )		SHIBBOLETH ( Version 0.7 )		TEQUILA ( Version 1.1 )	
Availability / Support			M/W	1-10	%	Description	Yes/No; 1-10	Description	Yes/No; 1-10	Description	Yes/No; 1-10
4.01	Availability	Software has to be available in a stable version	W	9	26%	in production since some time	6	First release after beta now available, not yet featured as designed	5	in development, based on GASPAR	4
4.02	Documentation	Documentation of design principles and software has to be available	W	7	20%	different Documentation available	7	Deployment guides available	7	available	7
4.03	Support	Support organization has to be in place	W	7	20%	authors support is available; building of a external support is in discussion	6	Mailing List with good responsiveness	4	support limited to one person	2
4.04	Current level of deployment	Software has to be deployed in a productive environment spanning more than one organization	W	6	17%	deployed in a productive environment, but mostly spanning one organization	3	Several test- and pilotsites/applications in the US	2	GASPAR deployed at EPFL	1
4.05	Licensing schema	Preferable Open Source with a licensing schema, which allows reuse and further development of software by Swiss AAI community	W	6	17%	Open Source	10	licensing issues with OpenSAML not yet clear	9	Open Source	10
				35	100%		6.4		5.4		4.7
<b>Overall Importance of groups of criteria</b>											
2.x	Ease of Integration		W	10	33%		2.5		2.7		2.4
3.x	Architecture, Security		W	10	33%		2.0		2.6		2.7
4.x	Availability, Support		W	10	33%		2.1		1.8		1.6
				30	100%		6.6		7.0		6.7