Requirements Discussion for the Blueprint Architecture

This section provide a run through the requirements (in the appendix) to see which ones should be fulfillable with by the AARC Blueprint Architecture and which ones remain still to be addressed.

Not all of the AAI solutions are currently in production use. Those of DARIAH, EGI, EUDAT and Umbrella are in production; as for the rest, their status is somewhere between prototype and pre-production. Regarding the non-production AAIs, it is therefore hard to determine which requirements are addressed in detail and which ones are not. However, from our coarse grained perspective, i.e. from the architectural point of view, we can reason about requirements being addressed or not.

To avoid addressing every single requirement individually, we have grouped the requirements into seven groups:

Requirements relating to Training (A7) are left out of this scope, since they can not be addressed from an architectural point of view.

A.1 Guest Identities and Levels of Assurance

- R2 Homeless Users
- R16 Social media identities
- R3 Different Levels of Assurance
- R13 Step up authentication
- R17 Integration with e-Government infrastructures

Topic A.1 concerns the integration of the so called social IdPs, such as Google or Facebook, in addition to the institutional accounts. This topic is investigated fully in JRA1.3 (“guest identities”) and deliverable MJRA1.2; what follows here is a brief discussion about the implications arising from the architectures discussed above.

The main problem encountered here is that all information about users is self asserted, i.e. not verified by an instance typically trusted by SPs. This has lead to systems like for example Umbrella, which understands the authentication simply as a unique and persistent identifier. With Umbrella or Google, when used as “low-assertion” identities, the ID-consuming sites are required to perform additional verification of the identity of the user, which happens on site via a passport inspection. Typically, additional attributes are associated with the identity. In general, (non-IdP) attributes are used for fine-grained authorisation such as group memberships, but for guest identities attributes can also describe the LoA associated with the external IdP - thus enabling multi-LoA infrastructures - or the LoA associated with a particular credential, as with reputations. In the latter case, while the infrastructure doesn’t fully trust all authentications from the guest IdP, it does assign higher trust to particular users who can be trusted to know what they are doing (which in turn is based on reputation or out-of-band knowledge). This is covered in the JRA1.3 deliverable, MJRA1.2. The bottom line of this is that additional AAs can be useful in enabling guest identities. In turn, enabling guest identities helps bringing in users in the “long tail” as well as younger researchers who grew up with social media and expect to continue to use them [FIM4R]. In particular, “homeless” users - users not backed by an institute in an existing federation - can be supported via guest identities, or via community-specific IdPs such as Umbrella ID. Thus, guest identities provide fulfil R2 (homeless users) and R16 (social media identities).

From our analysis it is clear, that there is no universal approach to address Levels of Assurance in the detail that we understand is required, today. This is also discussed in more details in MNA3.1 document.

- IGTF is the Interoperable Global Trust Federation, an activity which ensures that national science/research or NREN or commercial CAs are trusted on infrastructures around the world. Today, IGTF has four LoAs (“authentication profiles”) which CAs are required to follow as a minimum.
- The European Grid Initiative (EGI) and OpenScienceGrid (OSG) in the US agreed a common level of assurance, based on trusting IGTF plus a few additional IdPs where the LoA is not high enough to join the IGTF, or it was otherwise not practical to join the IGTF, but there is a common user base supporting it. Indeed, already in 2004, the forerunner projects, TeraGrid in the US and either EGEE or individual NGIs in Europe, negotiated mutually trusted CAs.
- NIST SP800-63 implements four levels of assurance, based on risks associated with misuse of credentials and other types of authentication errors, and used by US federal agencies - in particular they apply to the DoE laboratories with which European grids collaborate. However, the NIST levels were not reused directly as practical experiences with the early grids found that levels three and four (the highest) were too high, and in practice the IGTF levels are between NIST levels one and two.
- eIDAS regulation [Christos Kanellopoulos (grnet)?] In July 2015, European parliament established the eIDAS regulation whose implementing act 2015/1502 introduces three LoA levels for eID providers; low, substantial and high. eIDAS becomes effective in 9/2018 after which public organisations must support it in cross-national use of services requiring citizen authentication.
- The Kantara Initiative also defined four different levels of assurance. While based on the NIST SP800-63 levels, Kantara offers a framework around it - the Identity Assurance Framework - which defines criteria for assessment which enable participants to become certified against a particular LoA.

There exist standards (such as ISO/IEC 29115 or NIST SP800-63) which describe LoA using four levels, and also Vectors of Trust; however, there is no universal approach in comparing or exchanging LoA attributes between communities. There are examples where different communities are cooperating and successfully exchanging LoA attributes, but the process was not automatic and always involved an intervention on the community side (which may include, but is not limited to, describing a new LoA, or selecting an existing standard, e.g. NIST, eIDAS). Therefore, for the moment, this requirement (R3, but also R13 and R17) is mostly not used (ELIXIR will provide step-up authentication R13).

A.2 User Identification

- R9 Unique user identities
- R8 Persistent User Identifiers
- R10 User--managed identity information

User identifiers can generally be described by the following properties,:

- personal - denotes an identifier that is intended and designed for use by a single person, as opposed to shared (or guest) user accounts such as "libraryuser1@university.org". Based on NA3.1 (“AARC Policy Harmonisation - Level of Assurance”) work, personal identifiers are required by all research communities (and not delivered by all NREN feds).
- persistent - denotes a user identifier intended to be used for an extended period of time across multiple sessions. Conversely, an identifier intended to be used for a relatively short period of time that need not span multiple sessions is called a transient identifier.
- revocable - whether a given user identifier can be revoked. An identifier that persists over the entire lifetime of a user's relationship with an IdP is called a permanent identifier.
- non-reassignable - whether a given user identifier is assigned exclusively to a specific person, and, if the person ever stops using the identifier, is “remembered” in order to prevent reassignment to another person (e.g. with the same name). Based on the NA3.1 work, non-reassignable identifiers were another strong requirement from the research infrastructures.
- opaque - denotes a privacy-preserving user identifier that by itself provides no information about the user (e.g a UUID Version 4). An identifier that can be used to identify the subject is called a transparent identifier (e.g an email address).
- targeted - whether a given user identifier is intended for a specific relying party (or parties). A non-targeted identifier is a shared identifier.
- unique - whether a given user identifier is unique within the namespace of the identity provider and the namespace of the service provider(s) for whom the value is created.
    - Generally, uniqueness is interpreted as “there is a unique, individual, person behind this identifier,” and, in particular, every time the infrastructure sees the identifier, it can be sure it is the same person. In other words, there must be no accidental name clash (two different IdPs have assigned the same identifier to two different people - see also “global” property below), nor is there a deliberate name clash (two people are sharing the same credential, so necessarily present the same identifier to the infrastructure). In e-Infrastructures it is usually required that people do not share credentials with each other.
    - Uniqueness is a necessary feature for traceable identities (can be traced to a real-life identity), for assigning authorisation attributes to the identifier (by an AA), so this feature is desirable by many infrastructures.
    - Note that uniqueness does not imply non-reassignability, since the same unique identifier may be re-assigned to any given user at a certain point in time. Email address (such as givenname.surname@domain) is an example of a unique (in time) identifier which can be reassigned to another person (with the same name) once it is no longer used by the first person.
    - Another approach to uniqueness is to randomly generate the identifier; in this case global uniqueness cannot be guaranteed as another IdP may happen to generate the very same random identifier, but the probability of collision can be made negligibly small.
    - Yet another aspect of uniqueness is whether a given person can have more than one credential at the same time. Usually e-Infrastructures permit this, as it is a common case that people use one credential for testing and another for production, or they may have accounts with more than one institute, or may use their social media or external mail account in addition to their work account. As regards preventing users from having multiple accounts, one view is that, even if possible, it would be much more work to check whether the same person already has a credential when they apply for a new one - it would mean checking the identity associated with both the old and the new credential (or the account), potentially leaking data about the other person if they are different. Obviously it makes it harder to ban a user by name if they can use a different credential, so the extra effort of checking that the identity is truly unique - that there is only one identifier assigned to a given person, for that infrastructure - should be weighted against the requirement to ban users by name.
- global - whether a given user identifier is globally unique, i.e. intended to be unique across all instances of that attribute in any provider. Global uniqueness is typically ensured by using the identifier of the issuer as a qualifier (e.g. the DNS domain associated with the issuer). Email address is an example: as mail in general needs to be routed using globally consistent records maintained by the DNS systems, email address as an identifier needs to be globally unique (the exceptions being mail routed only locally such as @localhost).

See a comparison of some commonly known user identifiers based on the properties above in the table that follows:

Identifier	Persistent	Revocable	Reassignable	Opaque	Targeted	Globally Unique
SAML2 Transient NameID	No	N/A	N/A	Yes	N/A	Yes
SAML2 Persistent NameID	Yes	Yes	No	Yes	Yes	No
eduPersonTargetedID						
eduPersonPrincipalName	Yes	Yes	Yes	No	No	Yes
eduPersonUniqueId	Yes	Yes	No	Yes	No	Yes
eduPersonOrcid						
schacPersonalUniqueCode	Yes	Yes	No	No	No	No
schacPersonalUniqueID	Yes	Yes	No	No	No	Yes
OIDC public sub claim	Yes	Yes	No	Yes	No	No
OIDC pairwise sub claim	Yes	Yes	No	Yes	Yes	No

X.509 Certificate	Yes	Yes	Yes	No	Yes	Yes

The following requirements with regards to user identification were highlighted by the analysed communities:

- R8 - Persistent user Identifiers
- R9 - Unique user identities

eduPersonTargetedID (ePTID), eduPersonPrincipalName (ePPN) are currently in place.

ePPN is the equivalent of a username. As such, it has most of the properties typically associated with usernames (such as uniqueness and a naming convention of some sort), with the added property of global uniqueness through the use of a scope/qualifier. However, depending on the local or federation policies, the possibility for reassignment does still exist within the domain of a given identity provider.

ePTID is an alternative to ePPN as a user identifier. Theoretically speaking, an ePTID value is a tuple consisting of an opaque identifier for the principal (i.e. a user identifier), a name for the source of the identifier (i.e. the IdP), and possibly a name for the intended audience of the identifier. The audience typically refers to a single SP but it may also identify a group/collection of SPs. In practice, the ePTID may not contain the name of the audience but will just be (re)generated on the fly whenever the user needs to authenticate to a service in the audience. The ePTID value is persistent, non-reassignable, and unique (no other user has this id), but it is not unique in the stronger sense (the user may have more than one ePTID, see section A.2 for a discussion of “unique”). Note that the ePTID can remain consistent across multiple service providers when these providers have been grouped into "Affiliations" for policy purposes and need to share information about their users.

Based on the analysis above it is clear that the widely adopted ePPNs and ePTIDs cannot meet the requirement for a globally unique, shared, persistent and non-reassignable user identifier. Therefore, it is suggested to:

1. require from federations/IdPs a policy of non-reassignment for ePPNs, or
2. promote the use of eduPersonUniqueID (ePUID). An ePUID must be assigned so that no two values created by distinct identity systems could collide. This identifier is opaque and, once assigned, it cannot be reassigned to another principal. An ePUID identifier may be shared; however, since it can be used to identify an individual, it qualifies as personal data and therefore data protection laws apply. An ePUID value should remain stable over time, regardless of the nature of association (i.e. affiliation to the home organisation), interruptions in association, or complexity of association by the principal with the issuing identity system. When possible, the issuing identity system should associate any number of principals associated with a single person with a single value of this attribute. schacPersonalUniqueCode and schacPersonalUniqueID could also be considered. ORCID identifiers have similar properties, however the published eduPersonOrcid attribute specification is still under trial use and evaluation.

Finally, it should be noted that the use of the suggestions above does not prevent single individuals from having multiple globally unique, shared, persistent and non-reassignable identifiers in parallel. As previously mentioned, a practical consequence of this is that if a person with a given identifier is “blacklisted” in a research infrastructure, they can possibly register a new one (or use a different one) in return. The use of a government eID or schacPersonalUniqueID may be an exception in this case as it specifies a legal unique identifier for the subject it is associated with.

A.3 Attributes: Groups and Authorisation

- R4 Community based authorisation
- R5 Flexible and scalable attribute release policies
- R12 User groups and roles
- R_P_5 Semantically harmonized identity attributes

R4 (Community based authorisation), R12 (User groups and roles):

Attribute Aggregation models: Push and Pull models in Attribute Aggregation

While it is currently rare that users need attributes from more than a few (one or two) AAs, the need for combining attributes from more AAs may become more frequent - indeed, such a situation is desirable when we aim to foster cross-community collaborations and need to combine roles from different communities. The two models (push/pull) differ in how they handle the situation and scale with the number of AAs. For the push model, users will obtain the attributes they need and push them to the SP every time they use their credential. In the pull model each SP must obtain the credentials after successful user authentication. Both models have immanent pros and cons which are detailed in AARC MJRA1.3.

Technology capabilities for Attribute Aggregation

“The Grid” has implement a solution that works using VOMS, where a small number (tens) of VOMS servers support hundreds of VOs with thousands of users. However, the size of “the Grid” was tailored to fit the specific communities and it does not work well when trying to scale it to fit larger amounts of communities. Also, extending and opening the grid to other communities is difficult.

However, VOMS implements a commonly used, and well understood solution to working with community based AAI information. It is using “third party group or rights management”, which enables third parties (a community, a PI) to self manage e.g. a group by adding members to that group. This information can be encoded into attributes served by an Attribute Authority (AA). Attribute aggregation can be used by an SP to collect additional attributes about a user (such as the group membership).

Several challenges still exist however, especially in regard to scalability, since currently in SAML SPs have to preconfigure which Attribute Authorities they want to query; establishing trust between services and attribute authorities is not automated and services potentially have to query multiple AAs per user-login to ensure all groups are available. That said, the creating of trust between AAs and an SP appears more feasible than the home-IdP SP trust relation. Especially since for a community to access an SP, personal contacts need to be established first, while there is no personal contact with any home-IdP provider in that process.

From an architectural point of view, we distinguish between front and back channel attribute communication - the former being where attributes are passed along by, or with, the user’s authentication tokens. We also distinguish between push and pull, whether attributes are pushed by the user or pulled by the SP.

- VOMS is an example of front channel push: the user obtains their own attributes from a VOMS server, attaches them to their GSI proxy, and submits them to the service.
- OIDC includes a back channel pull mechanism, namely the 'userinfo' call, which enables an authorised server to fetch additional information about the user who has authenticated.
- Gridmap are lists of distinguished names (from X.509 certificates and proxies) along with their local account mappings. These files can be seen as a back channel push, as they are (sometimes) generated on an internal system and pushed out to the servers that will enforce them.

The following examples include a brief discussion of the attribute PUSH/PULL architectures; the reader is referred to the MJRA1.3 Milestone ("design for integration of an attribute management tool") for additional discussion.

R5 (Flexible and scalable attribute release policies), R_P_5 (Semantically harmonized identity attributes):

Token Translation Services (TTS) can be used to harmonise identity attributes. While a TTS may accept authentications from multiple external IdPs. It should ensure that the translated credential (attribute) is harmonised regardless of which external IdP was used. However, this is not always possible. One common approach is to add email address when it is not being published by the external IdP.

Another approach to harmonisation is to restrict it to a single IdP. Many projects (such as DARIAH, CLARIN) or communities (Umbrella ID) have set up a single IdP covering the relevant user base in order to ensure consistency. However, here the harmonisation problem arises again once communities start to collaborate or to use shared infrastructures.

A third, possibly simpler, approach is to mandate that all IdPs must publish a certain baseline set of attributes. However, current experience shows that this is not feasible, e.g. eduGAIN has struggled with this issue since the beginning but is currently moving towards giving up and instead only providing guidelines on how IdPs and SPs can agree on attributes. There are four identified issues:

- Attributes used for representing the same concept - some federation uses cn instead of displayname, some federations use edupersonscopedaffiliation instead of edupersonaffiliation
- Semantics of the attributes - variations in meaning for seemingly same attributes such as eduPersonAffiliation, the most widely used attribute to describe end user's role in their Home Organisation
- Other fundamental attribute properties, such as if eduPersonPrincipalName attribute values are re-assignable
- Privacy - the willingness of IdPs to release the needed attributes due to the privacy concerns

For example, all IdPs in a federation could be required by federation policy to publish eduPersonTargetedID and eduPersonScopedAffiliation, and perhaps leave additional attributes as optional or negotiated with SPs. However, once again harmonisation attributes arise when the identities are used outside of the original context (e.g. a national federation joining eduGAIN). Then it can become necessary to add a TTS. However, using TTS may not solve policy issues in regard to attribute release and attribute values (e.g. for eduPersonAffiliation, home organisation does not use ePA="faculty" but uses ePA=" employee" instead for both researchers and administration staff, this may cause the TTS to lack sufficient information on how to populate ePA="faculty" for proper persons).

A.4 Attributes: Release

- R_P_3 Sufficient Attribute release
- R6 Attribute Aggregation / Account Linking

A.5. Technology requirements

- R7 Federation solutions based on open and standards based technologies
- R14 Browser and non-browser based federated access
- R15 Delegation

R7 (Federation solutions based on open and standards based technologies):

More information is provided in MJRA1.1 deliverable: see e.g. sections 2.1 (SAML), 2.3 (OIDC), 2.5 (Moonshot), 3.26 (LDAP-Facade) and 2.7 (Kerberos). Coincidentally, these are also discussed in the following section on non-browser access to federated identities.

R14 Browser and non-browser based federated access:

The requirement for both browser and non-browser access to services arises because federated identity management is often implemented using only web browsers, relying on HTTPS protocols and in particular redirects to discover endpoints and carry messages between them. A typical example is the SAML WebSSO profile. The question then arises how one can support non-web access, e.g. secure shell (ssh) login or command line tools: while it is possible for a command line client to simulate a web client, the responses being returned from the server may require interactive updates or, having been designed to look attractive in a web browser, may be hard to parse. The SAML ECP (Enhanced Client or Proxy Profile) was designed to enable command line access (and similar) in an infrastructure with SAML SPs and IdPs. However, ECP is as of this writing (Apr 2016) not widely supported; the expected upgrade of Shibboleth IdPs to version 3 is expected to alleviate this situation since it should support ECP by default (and IdP version2 is scheduled to become unsupported by summer 2016).

In practical deployments, one often distributes ssh keys; by managing them as an attribute associated with the user, e.g. via LDAP or a web profile - even though ssh (in this case) uses public/private keys, this is not a full federated identity since the user authenticates directly to an SP rather than via an IdP, but it avoids the extra lookup at login time. While this is a simple and seemingly attractive solution, it needs a key synchronisation process at every SP, and in practice keys can grow "stale" (not deleted or updated if user's situation changes). It also leaves open the question of how the user's ssh key is uploaded to the LDAP server in the first place, as well as access control to the LDAP server.

Another alternative is to use X.509 certificates to authenticate: there are variants of ssh implementations that support X.509. This approach helps solve the key distribution problem that we saw with LDAP, since the server trusts the CA; however, many user communities find X.509 certificates hard to use, perhaps because of the inconsistent management in browsers, and command line tools for certificate management (openssl, java key store), while not always needed, are not widely known for their user friendliness. See also MJRA1.1, section 2.2.

Kerberos (and hence also Active Directory) supports command line access for a wide range of tools, but is usually restricted to a single realm in a single institute; and even if cross realm authentication is set up, it would have to be set up manually for each pair of realms, and even that is typically restricted to a single organisation.

API Access:

This is not a requirement collected in the requirement analysis, but it becomes evident, that RESTful APIs don't work well with SAML. One possible technology that can be included is OpenID-Connect (OIDC). OIDC is an authentication layer or a simple identity layer on top of the OAuth 2.0 framework, which allows clients to access resources (e.g. basic profile information) of a user while verifying identity based on the authentication performed by an authorisation server. Concretely, OIDC provides a standardised RESTful API, using JSON as a data format, therefore the delegated authorisation can be achieved in an interoperable manner. OIDC does not cover a use case of recursive or multiple-level delegation, this refers to delegating authorisation to one client to another client by the user.

R15 Delegation: delegation denotes the situation where one entity (delegator) authorises another entity (delegatee) to act on its behalf; this action is usually an interaction with a third party (a resource). Delegation can be achieved by "impersonation" i.e. handing over a token to the delegatee, who can carry out actions as the delegator. Different technologies offer different capabilities for controlling the actions of the delegatee.

SAML (i.e. user to SP A to SP B) is commonly being realized using the following pattern:

- First there must be a SAML authentication to SP A, e.g. using the WebSSO profile.
- Then SP A would need to play the role of a SAML ECP client, issuing a HTTP request for some resource X to SP B.
- Then SP B answers (to SP A) with a PAOS authentication request directed to the IdP.
- Then SP A takes the authentication request, and using its own original SAML token from the first step, authenticates at the IdP.
- Only then the IdP answers (to SP A) with the SAML response for SP B. SP A then forwards it using PAOS to SP A..
- In the end, SP B would deliver resource X to SP A.

It is actually possible to realize multiple-level delegation using this general pattern. However, the interaction pattern is seen as too complicated, and implementors usually resort to protocols like OIDC, which make simple delegation much easier.

R15 Delegation

The Federated AAI framework should provide the capability for the users to delegate third parties, mostly computational tasks or services, to act on their behalf. This allows users to run thousands of actions in parallel without the need for interactive access, for example to save output data.

A4.3.6 Other requirements

Here we list the requirements that are not discussed or targeted within the Blueprint Architectures:

Privacy, legal issues, and policies

- R_P_2 Federated Incident report Handling
- R18 Effective Accounting
- R11 Up to date identity information
- R_P_1 Policy Harmonization
- R_P_7 Best practices for terms and conditions

Training

- R_P_4 Awareness about R&E Federations
- R_P_6 Simplified process for joining identity federation
- R1 User and Service Provider friendliness