

**EGI-Engage**

Pilot services and best practices to enable federated AAI solutions released

M3.4

|  |  |
| --- | --- |
| **Date** | 06 June 2016 |
| **Activity** | JRA1.1 Authentication and Authorization Infrastructure |
| **Lead Partner** | GRNET |
| **Document Status** | DRAFT |
| **Document Link** | https://documents.egi.eu/document/XXX |

Abstract

This milestone document presents the pilot service developed along with the best practices developed in the first year of the EGI-Engage JRA1.1 activity.

**COPYRIGHT NOTICE**



This work by Parties of the EGI-Engage Consortium is licensed under a Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/). The EGI-Engage project is co-funded by the European Union Horizon 2020 programme under grant number 654142.

**DELIVERY SLIP**

|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Name*** | ***Partner/Activity*** | ***Date*** |
| **From:** | Christos Kanellopoulos | GRNET/JRA1.1 | 6/6/2016 |
| **Moderated by:** |  |  |  |
| **Reviewed by** |  |  |  |
| **Approved by:** |  |  |  |

**DOCUMENT LOG**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Issue*** | ***Date*** | ***Comment*** | ***Author/Partner*** |
| **v1.0** | 6/6/2016 | Initial version submitted for internal review | Christos Kanellopoulos/GRNETNicolas Liampotis/GRNET |

**TERMINOLOGY**

A complete project glossary is provided at the following page: <http://www.egi.eu/about/glossary/>

**Contents**

[1 Introduction 6](#_Toc453343255)

[2 Service architecture 6](#_Toc453343256)

[2.1 High-Level Service architecture 6](#_Toc453343257)

[2.2 Integration and dependencies 8](#_Toc453343258)

[2.2.1 Integration with SAML Identity/Service Providers 8](#_Toc453343259)

[2.2.2 Integration with Social Identity Providers 10](#_Toc453343260)

[2.2.3 Integration with Perun Attribute Authority 12](#_Toc453343261)

[2.2.4 Integration with GOCDB Attribute Authority 13](#_Toc453343262)

[2.2.5 Integration with CILogon-like Token Translation Service – End-Entity Certificates 14](#_Toc453343263)

[2.2.6 Integration with CILogon-like Token Translation Service – Per-User Sub-Proxies (PUSP) 16](#_Toc453343264)

[2.2.7 Integration with COmanage Registry – User Enrollment 17](#_Toc453343265)

[3 Release notes 18](#_Toc453343266)

[3.1 Requirements covered in the release 18](#_Toc453343267)

[4 Feedback on satisfaction 18](#_Toc453343268)

[5 Future plans 19](#_Toc453343269)

**Executive summary**

Task JRA1.1 started its activity in May 2015 focussing on collecting requirements from EGI users and key stakeholders, in order to understand their short-term and medium-term requirements, and establishing relationships with AARC, GN4, EUDAT2020 and PRACE, in order to work together towards an interoperable AAI. A liaison with the AARC project has been established to adopt AAI policies, solutions and best practices defined at European level and deal with problems that require a larger scope to be resolved, such as the lack of globally unique identifiers, levels of assurance etc.

The outcome of this process has been a list of core requirements that drove the design of new EGI AAI:

* Users should be able to access EGI services with credential released by his/her home organisation enabling the support for National Federation via eduGAIN.
* the so-called ‘homeless users’, who cannot rely on a reliable institution IdP shoud be also taken into account.
* Each supported IdP should release at least an identifier that uniquely identifies the user in the scope of that organization.
* Attributes should be extracted by different sources including community attribute providers.
* A Level of Assurance (LoA) should be associated to each identity in the EGI infrastructure scope.
* A persistent non-reassignable unique identifier for user is needed to manage the accounting linking.
* credential translator mechanisms/token translator services (TTSs) is needed to hide the complexity of the new EGI AAI to the service providers.

By the end of the first year of the JRA1.1 activity in the EGI-Engage project, the EGI AAI is fully functional in terms of core features and EGI has started on-boarding scientific communities. The recent introduction of the pilot CILogon service, enables all users to access even the legacy non-web EGI Services through the EGI AAI.

By the end of Q2 2016 it is expected that the EGI AAI will join eduGAIN as Service Provider supporting the GÉANT Data Protection Code of Conduct and the REFEDS Research & Scholarship entity category. Through eduGAIN, EGI Services will automatically become available to more than 2000 Universities and Institutes that are connected to the 38 eduGAIN Federations. Complementary to this, users without an account on a federated Identity Provider will be able to use their Google, Facebook, LinkedIn and ORCID accounts to access EGI Services that do not require substantial level of assurance.

In parallel, we are working on the first phase of the pilot with the EGI Competence Centres in order to connect them to the EGI AAI. This is an interactive process, which allows us to shape the EGI AAI exactly to the needs of our customer base.

In the third quarter of this year, we will continue with the second phase of the pilot, by the end of which we expect to have all the EGI scientific communities on board the EGI AAI. In addition, we will be introducing the new OpenID Connect interface, which will enable us to introduce new services to the EGI platform in a faster and friendlier way.

# Introduction

|  |  |
| --- | --- |
| **Tool name** | EGI AAI CheckIn Service |
| **Tool url** | https://aai.egi.eu/proxy |
| **Tool wiki page** | https://wiki.egi.eu/wiki/AAI |
| **Description** | Provides Authentication and Authorisation, enabling user-friendly and secure federated access to EGI services.  |
| **Value proposition** | The EGI AAI CheckIn Service enables research communities to access the EGI services without having to deal with X509v3 certificates. Researchers from home organizations that participate in one of the eduGAIN federations will be able to access the EGI services using the same credentials they are using at their home organization. Furthermore, the EGI AAI CheckIn Service supports user authentication with social media identities, enabling even those users who do not have a federated account at a home organization (such as many users that belong to the “Long Tail of Science”), to be able to access the EGI services in a seamless way without compromising the security of the EGI platform. The EGI AAI CheckIn service can connect to existing community based AAIs and it can be offered as an “Identity Access Management as a Service” to those communities, which do not have or do not want to operate their own AAIs. |
| **Customer of the tool** | NGI;RI;Resource Providers |
| **User of the service** | All EGI users |
| **User Documentation**  | https://wiki.egi.eu/wiki/AAI#Documentation |
| **Technical Documentation**  | https://wiki.egi.eu/wiki/AAI#Documentation |
| **Product team** | GRNET |
| **License** | Apache 2.0 License  |
| **Source code** | https://github.com/rciam |

# Service architecture

## High-Level Service architecture

The AAI activity in EGI-Engage started in May 2015. During the first months of the project, we worked together with the AARC project in order to identify the requirements of the scientific communities. This work resulted in a set of guiding principles:

* Users should be able to access the EGI Services using the credentials they have got from their Home Organizations using eduGAIN when possible, but alternative methods should be available
* EGI should expect to receive at least an identifier that uniquely identifies the user coming from within the scope of the authentication source.
* Within the EGI environment, a user should have one persistent non-reassignable non-targeted unique identifier.
* EGI should define a set of minimum mandatory attributes, without which a user cannot exist within the EGI environment.
* EGI should attempt to retrieve these attributes from the user’s Home Organization. If this is not possible, then an alternative process should exist in order to acquire and verify the missing user attributes.
* There should be a distinction (LoA) between self-asserted attributes and the attributes provided by the Home Organization/VO
* Access to the various services should be granted based on the VO/EGI roles the user has.
* EGI Services should not have to deal with the complexity of multiple IdPs/Federations/Attribute Authorities/technologies. This complexity should be handled centrally and should be hidden from the EGI Services.

Based on these principles and following the guidelines from the AARC project, we designed an architecture for the EGI AAI and a roadmap in order to incrementally introduce the new service elements on the EGI platform.



Figure 1: EGI AAI CheckIn Service High Level Architecture

The core of EGI AAI CheckIn Service is the IdP/SP Proxy component, which acts as a bridge between the EGI services and external authentication sources and identity providers. This separation between the internal services and the external authentication sources/identity providers allows the service developers to focus on the service features and not have deal with the complexity of multiple IdPs, Federations, Attribute Authorities and different authentication and authorization technologies. This complexity is “outsourced” and handled centrally by the proxy. Services need to established trust with just one entity, the IdP/SP proxy. Typically, services will have one static configuration for the IdP/SP proxy. Having one configured IdP, removes also the requirement from the services to operate their own IdP Discovery Service, which is a common requirement for services supporting federated access. Furthermore, all internal services will get consistent and harmonized user identifiers and attributes, regardless of the home organization or the research community that the user belongs to. Finally, this separation allows the change management of the internal services to not have dependencies on the change management cycles of the IdPs at the home organizations. IdPs establish trust with one entity, the operator of the IdP/SP proxy and they are not impacted by the change operations of each individual service.

## Integration and dependencies

### Integration with SAML Identity/Service Providers

This is the foundation of EGI AAI integration use case that enables users to access a web-based EGI service (Service Provider - SP) using their credentials and attributes from their home organisation (Identity Provider - IdP). The most common way for enabling federated access on web-based EGI services is implemented through the use of SAML. The typical Single Sign-On (SSO) flow begins with the user accessing an EGI application through their web browser (SP-initiated SSO), as depicted in the following diagram.



 **EGI SP flow:**

1. The user visits the web-based SP using her browser
2. The user selects to login using her federated account
3. The portal generates a SAML request and redirects the user’s browser to the SAML endpoint of the EGI IdP Proxy, embedding the SAML request in the URL
4. The user’s browser redirects to the EGI IdP Proxy and passes along the SAML request as a URL parameter

**EGI AAI IdP Proxy flow:**

1. The EGI AAI IdP Proxy verifies the SAML request, presents a set of Home Organisations (IdP Discovery Service) to the user and the user selects her Home Organization
2. The EGI AAI IdP Proxy generates a new SAML request and redirects the user’s browser to the SAML endpoint of the Home IdP of the user
3. The user’s browser redirects to the Home IdP and passes along the SAML request as a URL parameter
4. The user authenticates herself at the Home IdP and upon successful authentication the Home IdP builds a SAML assertion representing the user's logon security context, it is digitally signed and placed within a SAML message. The message is then placed within an HTML FORM as a hidden form control named SAML response.
5. The Home IdP sends the HTML form back to the browser in the HTTP response. For ease of use purposes, the HTML FORM is accompanied by script code that will automatically post the form to the destination site
6. The browser, due either to a user action or execution of an “auto-submit” script, issues an HTTP POST request to send the form to the EGI AAI IdP Proxy.
7. The EGI AAI IdP Proxy decrypts and verifies the SAML Assertion.
	1. The SAML Assertion must include at least one of the following: eduPersonUniqueId (ePUID), eduPersonPrincipalName (ePPN), or eduPersonTargetedID (ePTID).
	2. The EGI AAI IdP Proxy replaces the original user identifier with an ePUID which is generated by hashing the [ePUID|ePPN|ePTID] and is scoped at “egi.eu”.
	3. The EGI AAI IdP Proxy builds a new SAML assertion representing the user's logon security context, it is digitally signed and placed within a SAML message. The message is then placed within an HTML FORM as a hidden form control named SAML response.

**SP “Termination” flow:**

1. The EGI AAI IdP Proxy asks for user consent and when it is given, it sends the HTML form back to the browser in the HTTP response. For ease of use purposes, the HTML FORM is accompanied by script code that will automatically post the form to the destination site
2. The browser, due either to a user action or execution of an “auto-submit” script, issues an HTTP POST request to send the form to the SP.
3. The SP decrypts and verifies the SAML assertion and make an access check to establish whether the user has the correct authorisation to access the resource.
4. If the access check passes, the resource is then returned to the browser.

Based on the flows above, the following SAML IdP/SP entities are currently interconnected following the best practices and guidelines for interconnecting IdPs[[1]](#footnote-1) and SPs[[2]](#footnote-2):

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **AAI Software** |
| EGI AAI IdP/SP Proxy | IdP/SP Proxy | SimpleSAMLphp[[3]](#footnote-3) |
| GRNET VHO | IdP | Shibboleth[[4]](#footnote-4) |
| EGI SSO | IdP | Shibboleth |
| GOCDB | SP | Shibboleth |
| AppDB | SP | Shibboleth |
| ELIXIR AAI | IdP Proxy | OpenConext[[5]](#footnote-5) |

### Integration with Social Identity Providers

This is an extension to the above integration use case allowing users to authenticate against commonly used Social Identity Providers. As these providers do not provide SAML-compliant authentication mechanisms, the EGI AAI acts as an OpenID Connect/OAuth2-to-SAML bridge. In this context, it is required to map social identity profiles into SAML attribute assertions. These mappings are described hereafter. It should be noted that only user information, which is relevant to the REFEDS Research and Scholarship (R&S[[6]](#footnote-6)) attribute bundle, is covered in this section.

#### Google / OpenID Connect

Google's OAuth 2.0 APIs can be used for both authentication and authorisation. This OAuth 2.0 implementation conforms to the OpenID Connect (OIDC) specification and is OpenID Certified. As such, when including the OpenID scope, information about the user can be retrieved from the UserInfo endpoint in OpenID Connect format. The Claims[[7]](#footnote-7) returned in the UserInfo Response can be mapped to SAML attributes as follows:

|  |  |
| --- | --- |
| **Google (OIDC) user claim** | **EGI Profile SAML attribute** |
| sub | ePUID (scoped @google.com) → hashed ePUID (scoped @egi.eu) |
| name | displayName |
| given\_name | givenName |
| family\_name | sn |
| email | mail |

#### Facebook

Facebook allows retrieving user information through the /{user-id} Graph API endpoint[[8]](#footnote-8), following an OAuth 2.0 flow for authentication and authorisation. The returned fields of the Facebook user profile can be mapped to SAML attributes as follows:

|  |  |
| --- | --- |
| **Facebook user field** | **EGI Profile SAML attribute** |
| third\_party\_id (an anonymous, but unique identifier for the person that can be shared with third parties) | ePPN (scoped @facebook.com) → hashed ePUID (scoped @egi.eu) |
| id (this ID is unique to each app and cannot be used across different apps) | ePTID (“http://facebook.com!” + id) (omitted from the EGI identity attribute profile since the included ePUID is generated from the ePPN above) |
| name | displayName |
| first\_name | givenName |
| last\_name | sn |
| email | mail |

#### LinkedIn

LinkedIn relies on the OAuth 2.0 protocol for enabling authenticated access to its REST APIs that provide access to member data. More specifically, following a three-legged OAuth2 flow, LinkedIn user profile[[9]](#footnote-9) information can be accessed through the /people/~ REST API endpoint. The returned user fields can be mapped to SAML attributes as follows:

|  |  |
| --- | --- |
| **LinkedIn user field** | **EGI Profile SAML attribute** |
| id (a unique identifying value for the user, which is linked to the specific application) | ePTID (“http://linkedin.com!” + id) → hashed ePUID (scoped @egi.eu) |
| formatted-name | displayName |
| first-name | givenName |
| last-name | sn |
| email-address | mail |

### Integration with Perun Attribute Authority

The EGI AAI proxy has been integrated with Perun in order to retrieve information describing the user’s VO/group memberships. This information is encapsulated in URN-formatted eduPersonEntitlement values, which are incorporated into the original SAML attribute assertion sent by the user’s IdP before being passed on to the relying party. The interactions among the involved components have been visualised in the following diagram.



**Perun flow:**

1. The EGI AAI IdP Proxy makes a back channel LDAP query to the Perun service passing along the user identifier
2. Perun returns the group membership information in the LDAP query response
3. The EGI AAI IdP Proxy encapsulates the returned information in an eduPersonEntitlement attribute which is added to the SAML Attribute Response

### Integration with GOCDB Attribute Authority

The EGI AAI proxy has been integrated with the GOCDB in order to retrieve information describing the user’s role(s) on various entities, such as EGI sites. This information is encapsulated in URN-formatted eduPersonEntitlement values, which are incorporated into the original SAML attribute assertion sent by the user’s IdP before being passed on to the relying party. The interactions among the involved components have been visualised in the following diagram.



1. The EGI IdP Proxy makes a back channel REST API call to GOCDB[[10]](#footnote-10) passing along the user’s identifier contained in the SAML authentication response received by the IdP.
2. GOCDB returns the user roles in the HTTP response
3. The EGI IdP Proxy adds the received attributes to the SAML Attribute Response

### Integration with CILogon-like Token Translation Service – End-Entity Certificates

The EGI AAI Proxy has been integrated with a Token Translation Service (TTS), which is based on the CILogon[[11]](#footnote-11) model in order to enable federated access to EGI resources that require conventional identity and attribute certificates. CILogon is an open source project used to provide x509 certificates based on the authenticated user's federated identity. The TTS service that has been integrated with the EGI AAI, as in the case of CILogon, is based on OAuth for MyProxy (OA4MP[[12]](#footnote-12)). More specifically, the CILogon-like TSS service comprises the following components:

* OA4MP: OAuth 2.0 implementation
* Shibboleth: Service Provider 2.0
* MyProxy Server[[13]](#footnote-13)
* SimpleCA[[14]](#footnote-14)

The components of the CILogon-like TTS service and their interconnections with the EGI AAI have been depicted in the following diagram[[15]](#footnote-15).



**User flows:**

1. User goes to VO portal
2. browser redirect to /authorize endpoint on Master Portal
3. browser redirect to /authorize endpoint on Delegation Service (DS)
4. browser redirect (SAML) to EGI AAI IdP Proxy
5. user authenticates at their Home IdP through the EGI IdP Proxy (via WAYF)
6. redirect back (SAML) to /authorize on DS
7. redirect back to ‘redirect\_uri’ on Master Portal

(Master Portal retrieves access\_token-2 and uses it to obtain userinfo)

1. redirect back to ‘redirect\_uri’ on VO portal

**The next steps are all hidden from the user:**

1. VO portal retrieves access\_token-1 from Master Portal
2. VO portal uses access\_token-1 to obtain userinfo from Master Portal
3. VO portal calls /getproxy endpoint (on Master Portal) using access\_token-1, optionally with VOMS information (VO FQANs and/or VOMSES)
4. Master Portal checks presence of long lived proxy in MyProxy credential store, using MyProxy INFO command

In case there is no proxy yet:

* 1. Master Portal creates keypair + CSR
	2. Master Portal calls /getcert on DS using access\_token-2 and CSR
	3. Delegation Service (DS) does a MyProxy GET request at online CA, using the CSR
	4. online CA signs the CSR and returns the end-entity certificate to DS
	5. DS returns certificate to Master Portal
	6. Master Portal uses end-entity certificate and key to delegate (MyProxy PUT) a long-lived proxy to the MyProxy credstore
1. Master Portal retrieves short lived (VOMS) proxy from the MyProxy credstore
2. Master Portal returns proxy to VO portal

### Integration with CILogon-like Token Translation Service – Per-User Sub-Proxies (PUSP)

The steps here are almost identical to those presented in Section 2.2.5, except a (Per-User Sub) proxy is returned from the online CA instead of an end-entity certificate, which can then be stored directly. Details on the MyProxy reconfiguration can be found at <https://wiki.nikhef.nl/grid/PUSP_from_MyProxy>

**User flows:** E1-E8 (refer to Section 2.2.5)

**The next steps are all hidden from the user:**

1. VO portal retrieves access\_token-1 from Master Portal
2. VO portal uses access\_token-1 to obtain userinfo from Master Portal
3. VO portal calls /getproxy endpoint (on Master Portal) using access\_token-1
4. Master Portal checks presence of long lived proxy in MyProxy credential store, using MyProxy INFO command

In case there is no proxy yet:

* 1. Master Portal creates keypair + CSR
	2. Master Portal calls /getcert on DS using access\_token-2 and CSR
	3. DS does a MyProxy GET request at ‘online CA’, using the CSR
	4. online CA signs the CSR using the robot private key and returns a PUSP chain to DS
	5. DS returns proxy chain to Master Portal
	6. Master Portal uses proxy chain with the previously generated private key to store (MyProxy STORE) the long-lived proxy to the MyProxy credstore
1. Master Portal retrieves short lived proxy from the MyProxy credstore
2. Master Portal returns proxy to VO portal

### Integration with COmanage Registry – User Enrollment

The EGI AAI Proxy has been integrated with the COmanage Registry[[16]](#footnote-16) to provide a seamless onboarding experience for new EGI users and to support advanced account management features, such as the ability to link user accounts from different identity providers into a single EGI profile. This section puts the focus on the user enrollment flow, which has already been finalised, whereas the account linking process is still under development.

The COmanage Registry comprises a database for maintaining account information, as well as a Web UI and a rich set of APIs enabling federated access (SAML) to the user attributes managed by the COmanage Registry. To allow for the automatic registration of new users upon accessing an EGI service for the first time, the COmanage registry has been connected with the EGI AAI Proxy as both an Attribute Authority (AA) and a Service Provider (SP). Effectively, the flow is an extension of the baseline scenario presented in Section 2.2.1. More specifically, the final step (Step B7) becomes as follows:

1. The EGI AAI IdP Proxy decrypts and verifies the SAML Assertion.
	1. The SAML Assertion must include at least one of the following: eduPersonUniqueId (ePUID), eduPersonPrincipalName (ePPN), or eduPersonTargetedID (ePTID).
	2. The EGI AAI IdP Proxy replaces the original user identifier with an ePUID which is generated by hashing the [ePUID|ePPN|ePTID] and is scoped at “egi.eu”.
	3. The EGI AAI IdP Proxy queries the COmanage Registry (AA) using the ePUID from step B7b.

*If the user is found in the COmanage Registry:*

* + 1. The EGI AAI IdP Proxy builds a new SAML assertion representing the user's logon security context, it is digitally signed and placed within a SAML message. The message is then placed within an HTML FORM as a hidden form control named SAMLResponse.

*Alternatively, if the user is not found in the COmanage Registry:*

1. User's browser is redirected to the new user registration page in COmanage (SP)
2. The registration page gets populated with the user’s generated EGI unique identifier and all attributes asserted by the Home IdP
3. User fills in any mandatory attributes that may be missing, such as name or email address information.
4. User explicitly consents to the Terms of Use of the EGI AAI SP Proxy and then chooses to submit the user registration form.
5. COmanage sends confirmation link to the email address associated with the user’s EGI account
6. User opens the link in the email sent by COmanage and is navigated to the registration page to confirm ownership of the email address
7. User confirms email address through the COmanage registration page
8. User re-authenticates at their Home IdP (through the EGI AAI IdP Proxy) to finalise the new EGI user enrolment process.

# Release notes

## Requirements covered in the release

The EGI AAI CheckIn Service is still under active development and the first “production release” is expected by the fall of 2016. As this is a high impact service, the existing EGI Pilot is already operated in terms of a production service, with the difference that new features/changes are pushed on a daily basis. With the first production release, the service will enter also in the production change management cycle as the rest of the EGI services.

Information about the requirements covered and the features that have been implemented or which are in the plan until the first production release can be found in the EGI AAI CheckIn Service development roadmap[[17]](#footnote-17).

# Feedback on satisfaction

Currently the EGI AAI CheckIn Service is in the pilot phase. We have started the integration of EGI Services and Tools and the on-boarding of the Research Communities from the EGI Competence Centres. On the side of the EGI Service and Tools, GOCDB, AppDB and GGUS have already been integrated and we have started the work on the Federated Cloud Services. The choice of the services is driven by the use cases of the Research Communities that are being on-boarded. We have started with the research communities from the ELIXIR Competence Centre and we will continue with the research communities of the rest of the Competence Centres.

Between September and November, we are going to perform an assessment of the feedback received from the integration and on-boarding activities and we will proceed with a technology assessment and roadmap evaluation.

# Future plans

The immediate future plans of the EGI AAI CheckIn service can be found in the EGI AAI CheckIn Service development and integration roadmap. The longer-term plans of the service are going to be defined after the technology assessment.

1. https://wiki.egi.eu/wiki/AAI\_guide\_for\_IdPs [↑](#footnote-ref-1)
2. https://wiki.egi.eu/wiki/AAI\_guide\_for\_SPs [↑](#footnote-ref-2)
3. <https://simplesamlphp.org/> [↑](#footnote-ref-3)
4. <https://shibboleth.net/> [↑](#footnote-ref-4)
5. <https://openconext.org/> [↑](#footnote-ref-5)
6. <https://refeds.org/category/research-and-scholarship> [↑](#footnote-ref-6)
7. <https://openid.net/specs/openid-connect-basic-1_0.html#StandardClaims> [↑](#footnote-ref-7)
8. <https://developers.facebook.com/docs/graph-api/reference/v2.6/user> [↑](#footnote-ref-8)
9. <https://developer.linkedin.com/docs/fields/basic-profile> [↑](#footnote-ref-9)
10. <https://wiki.egi.eu/wiki/GOCDB/PI/get_user_method> [↑](#footnote-ref-10)
11. <http://www.cilogon.org/> [↑](#footnote-ref-11)
12. <http://grid.ncsa.illinois.edu/myproxy/oauth/> [↑](#footnote-ref-12)
13. <http://grid.ncsa.illinois.edu/myproxy/ca/> [↑](#footnote-ref-13)
14. <http://toolkit.globus.org/toolkit/docs/latest-stable/simpleca/> [↑](#footnote-ref-14)
15. A more detailed view of the CILogon-like TTS service architecture is available at <https://wiki.nikhef.nl/grid/CILogon_Pre-Pilot_Work#Detailed_Architecture> [↑](#footnote-ref-15)
16. <https://spaces.internet2.edu/display/COmanage/> [↑](#footnote-ref-16)
17. https://wiki.egi.eu/wiki/EGI-Engage:TASK\_JRA1.1\_Authentication\_and\_Authorisation\_Infrastructure#Development\_Roadmap [↑](#footnote-ref-17)