User Delegation in the CLARIN Infrastructure

Jonathan Blumtritt[1] **Willem Elbers**[2] **Twan Goosen**[2]
Marie Hinrichs[3] **Wei Qiu**[3] **Mischa Sallé**[4]
Menzo Windhouwer[5]
[1] Cologne Center for eHumanities – University of Cologne, [2] CLARIN ERIC,
[3] University of Tübingen, [4] NIKHEF, [5] The Language Archive – Meertens Institute
jonathan.blumtritt@uni-koeln.de, willem@clarin.eu,
twan@clarin.eu, marie.hinrichs@uni-tuebingen.de,
wei.qiu@uni-tuebingen.de, msalle@nikhef.nl,
menzo.windhouwer@meertens.knaw.nl

Abstract

The CLARIN research infrastructure aims to place language resources and services within easy reach of the humanities researchers. One of the measures to make access easy is to allow these researchers to access them using their home institutions credentials. However, the technology used for this makes it hard for services to make delegated call, i.e., a call on behalf of the researcher, to other services. In this paper several use cases, e.g., interaction with a researcher's private workspace or protected resources, show how user delegation would enrich the capabilities of the infrastructure. To enable these use cases various technical solutions have been investigated and some of these have been used in pilot implementations of the use cases. This paper reports on the use cases, the research and the implementation experiences.

1 Introduction

The topic of this paper is the interaction between two of the pillars of the CLARIN research infrastructure:[1] ease of access and integration of services. Ease of access has been implemented by enabling researchers to use their home institution credentials to access resources, tools and services offered by CLARIN on the web. This works well in many cases, but has turned out problematic for the cases where these services themselves need to access other services or resources on behalf of the researcher. To research possible solutions and implement them for a specific use case CLARIN-NL[2] has teamed up with the Dutch BiG Grid project.[3] Last year also a CLARIN-D[4] use case has been solved using the same solution and new CLARIN(-D) use cases are under investigation and in actual development. This paper reports on the results of the research and implementation of these different use cases.

The structure of the paper is as follows: in Section 2 it starts with a description of the problem, the requirements for a good solution, the possible solutions investigated and briefly mentions new development since the research was done. Section 3 then describes in depth the chosen solution and a first implementation thereof. Several use cases in the CLARIN infrastructure would profit from user delegation. These use cases and, where possible, experiences obtained during the implementation are described in Section 4. The paper ends with a description of future work and some conclusions.

[1] http://clarin.eu/content/mission
[2] http://www.clarin.nl/
[3] http://www.biggrid.nl/
[4] http://de.clarin.eu/

2 Shibboleth and User Delegation

Shibboleth[5] is the underlying technology that enables users to use the credentials of their home institute in the CLARIN infrastructure. It is based on the Security Assertion Markup Language (SAML; Cantor, 2012), as a Single Sign-On (SSO) system. Shibboleth is widely used in the research world,[6] providing single sign-on for web applications based on national federations, where the universities and research institutions function as Identity Providers (IdPs). The CLARIN centers that offer services, fulfilling the role of Service Providers (SPs), have grouped together in a CLARIN federation, which makes it administratively easy for the IdPs to deal with the CLARIN SPs.

Its wide support has made Shibboleth a good starting point for CLARIN, but it also has disadvantages. Shibboleth is typically aimed at users logging in and interacting with the SPs via their browser. Although the use cases described in this paper always start out in a browser session, the service invoked needs to invoke another service on behalf of the researcher. Shibboleth does not support this by default. In the next section possible solutions to enable such functionality are described.

2.1 Possible solutions

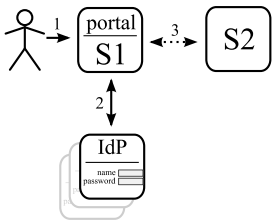
In the research phase of the CLARIN-NL/BiG Grid collaboration many solutions were considered and evaluated against the following requirements (grouped from 3 angles):

1) For the *User*:
   a) Single-Sign-On
   b) Access public and private services from within a portal (and other services)
   c) Transparent use, no required confirmation for every service or service access
2) For *Services*:
   a) Authentication by identity provider
   b) Authorization by service provider
   c) Nested service invocation possible (delegation)
   d) Easy to set up (for researcher)
3) For the *System* as a whole:
   a) Multi-federation authentication using SAML2
   b) REST and possibly SOAP
   c) Using proven technologies
   d) Operational effort minimal
   e) In-line with standards & best practices[7]
   f) Can we start today?

In this section the considered solutions and their evaluations are briefly discussed, for a more extensive discussion see Van Engen and Sallé (2011). In the descriptions and figures S1 indicates the service that calls another service, which is called S2, on behalf of the researcher (represented by the stick figure) authenticated by an IdP. Numbered arrows indicate subsequent requests between the parties involved.

Open

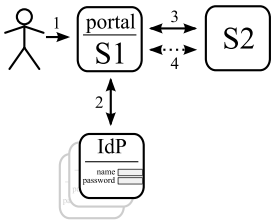
In this simple model all services trust each other. S1 includes the user identity with its request to S2, which accepts this without further checking. This is easy to setup, but does not scale up to the CLARIN infrastructure.



[5] http://www.internet2.edu/shibboleth/

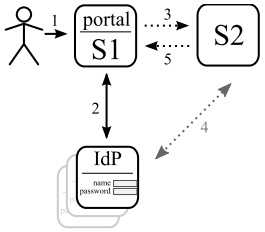
[6] See for example the coverage of research and education identity federations at https://refeds.org/index.html

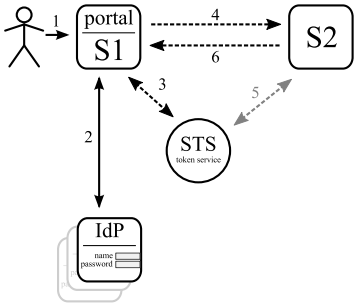
[7] This includes the requirement that the solution should be secure.



OAuth 1 (Hammer-Lahav, 2010)
This protocol is popular on the Internet and uses delegated security tokens for one site to access another site, e.g., allow LinkedIn to access one's Google address book. When S1 wants to access S2 the researcher's browser will be redirected to S2. There the researcher allows the access, and is redirected back to S1. The drawback is the need for separate confirmation for each combination of services.

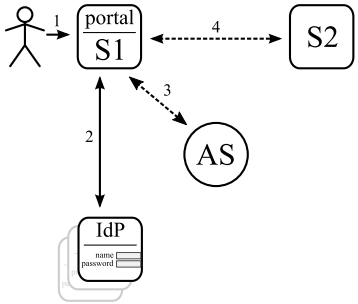
SAML ECP (SAML V2.0 Contributors, 2005)
Enhanced Client or Proxy (ECP) is developed to support SAML for programs other than the browser. For Shibboleth, it is actually supported but not enabled by default, while SimpleSAMPLphp[8] does not support delegation via ECP. SAML ECP therefore is not a viable solution: CLARIN cannot force the IdPs to enable ECP and furthermore, since ECP would require a configuration for each AP at each IdP, such a solution does not scale.

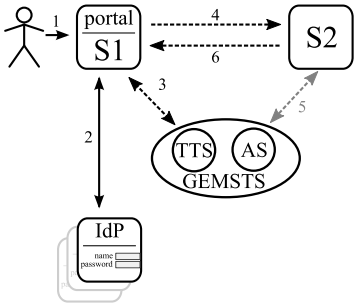




WS-Trust[9]
WS-Trust defines the concept of a security token service for SOAP web services. It is a flexible but rather complex setup, and can also be problematic for REST services.

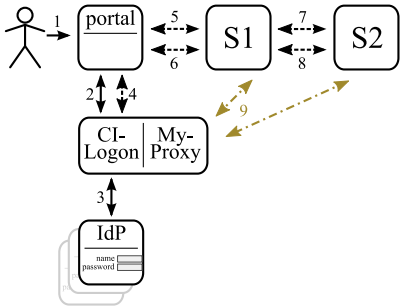
OAuth 2[10] *(Hardt, 2012)*
This next evolution of OAuth supports more scenarios. As in the WS-Trust case a central service, an Authorization Service (AS), allows S1 to request a security token to pass on to S2, which can check the validity of the token and receive the user identity. Although this solution was fairly new at the time, it was selected as the primary option to be further investigated. It has since then quickly become the de-facto authorization standard on the internet and is replacing OAuth 1.





GEMBus STS
The GEMBus framework[11] is intended as a multi-domain communication environment and provides a number of services, including a security token service. At the time of evaluation GEMBus was alpha software.

X.509 certificates (Cooper, Santesson, Farrell, Boeyen, Housley, & Polk, 2008)
These certificates are the basis of the widely used SSL and TLS protocols. They are based on a public key infrastructure where trusted certificates are signed by trusted certificate authorities (CA). Delegation can be implemented using proxy certificates and is used as such in the 'grid world'. At the cost of additional setup the, much feared, burden of managing the certificate/keypair can be hidden from the user. This solution was selected as the secondary option to be investigated in case the OAuth 2 solution would fail.



[8] https://simplesamlphp.org
[9] http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/ws-trust.html
[10] http://oauth.net/
[11] http://geant3.archive.geant.net/Research/Multidomain_User_Application_Research/Pages/GEMBus.aspx

2.2 Chosen solutions

Eventually two different solutions were chosen for further analysis, since they both could satisfy all the requirements. Firstly a solution based on OAuth 2 was chosen. The only identified risk to this solution at the time was the relative immaturity of OAuth 2 as a protocol and hence also of its implementations, since at the time, most (commercial) internet sites were still using the incompatible predecessor OAuth 1 protocol. This was therefore also the primary reason for choosing a second solution for further investigation. This second option, a solution based on X.509 certificates, which should then be used in such a way that they are hidden from the end-users, also could satisfy all the requirements, and most building blocks were already available at the time. Also this second solution has become more interesting over the past years, in particular in the scientific communities. All the other investigated options showed important shortcomings.

Hence it was decided to start with an OAuth 2 based proof-of-concept implementation, and depending on the experiences from that, to decide whether the X.509-based second option should be implemented as well.

2.3 New developments

Since the research reported on in Section 2.1 and the implementations efforts in the remainder of this paper the EUDAT project[12] has been investigating and developing a solution, named B2ACCESS, that is able to connect the different AAI infrastructures used within different communities, typically providing identity information, to the services offered within the EUDAT infrastructure. The solution provided by the UNITY software[13] supports this integration with different technologies such as SAML, OpenID, username/password and more. This allows for the authentication of the user using their federated identities and mapping these to an EUDAT identity which is then exposed to the EUDAT services in one of three ways: (1) X.509 certificates, (2) OAuth 2 and (3) SAML. Because of this flexibility this solution is very interesting since it allows for different options in the backend. There is support for OAuth 2, which is discussed in depth in this paper, but there is also support for X.509 certificates which might be a good candidate in specific scenarios. Although there is also SAML support, the limitations for the ECP support discussed earlier prevent this from being a viable alternative.

3 Configuring and Running an OAuth 2 Authentication Service

Figure 1 sketches the OAuth 2 delegation workflow in more detail: A user is logged in to Service 1 (S1), which is secured via a Shibboleth SP, using the IdP of his home institution. When the user triggers an action on S1 that requires access to a resource on Service 2 (S2), S1 redirects the user to the AS to collect an access token. Since the AS is also secured via an SP, it sends the user to the Discovery Service (DS) where he selects the IdP for authentication. The AS creates an authorisation code which is sent to S1 via the user. S1 uses it to request an OAuth 2 access token from the same AS. S1 then passes this access token to S2, which checks the validity of the token with the AS and receives user attributes in return (such as the user ID derived from the EPPN (*EduPersonPrincipalName*)). If the token is valid and S2 authorizes the user for the resource (a decision based on the user ID), S2 sends back the response to S1, which can then process it and complete the action triggered by the user. For the lifetime of the initial token, further communication between S1 and S2 can occur without the need to request another token.

In a second report, Van Engen and Sallé (2013) describe how, after attempts to use OAuth 2Lib,[14] a working solution was obtained using the *ndg_oauth* Authorization Server[15] combined with OAuth for Spring Security.[16] The *ndg_oauth* AS is implemented in Python, and for production it is advised to run it via WSGI in an Apache HTTP server. To get it to work for the use cases described below, i.e., to allow S2 to actually receive the user identity, some fixes were needed.

[12] http://www.eudat.eu
[13] http://unity-idm.eu/
[14] http://www.rediris.es/oauth2/
[15] https://github.com/cedadev/ndg_oauth
[16] http://projects.spring.io/spring-security-oauth/docs/oauth2.html

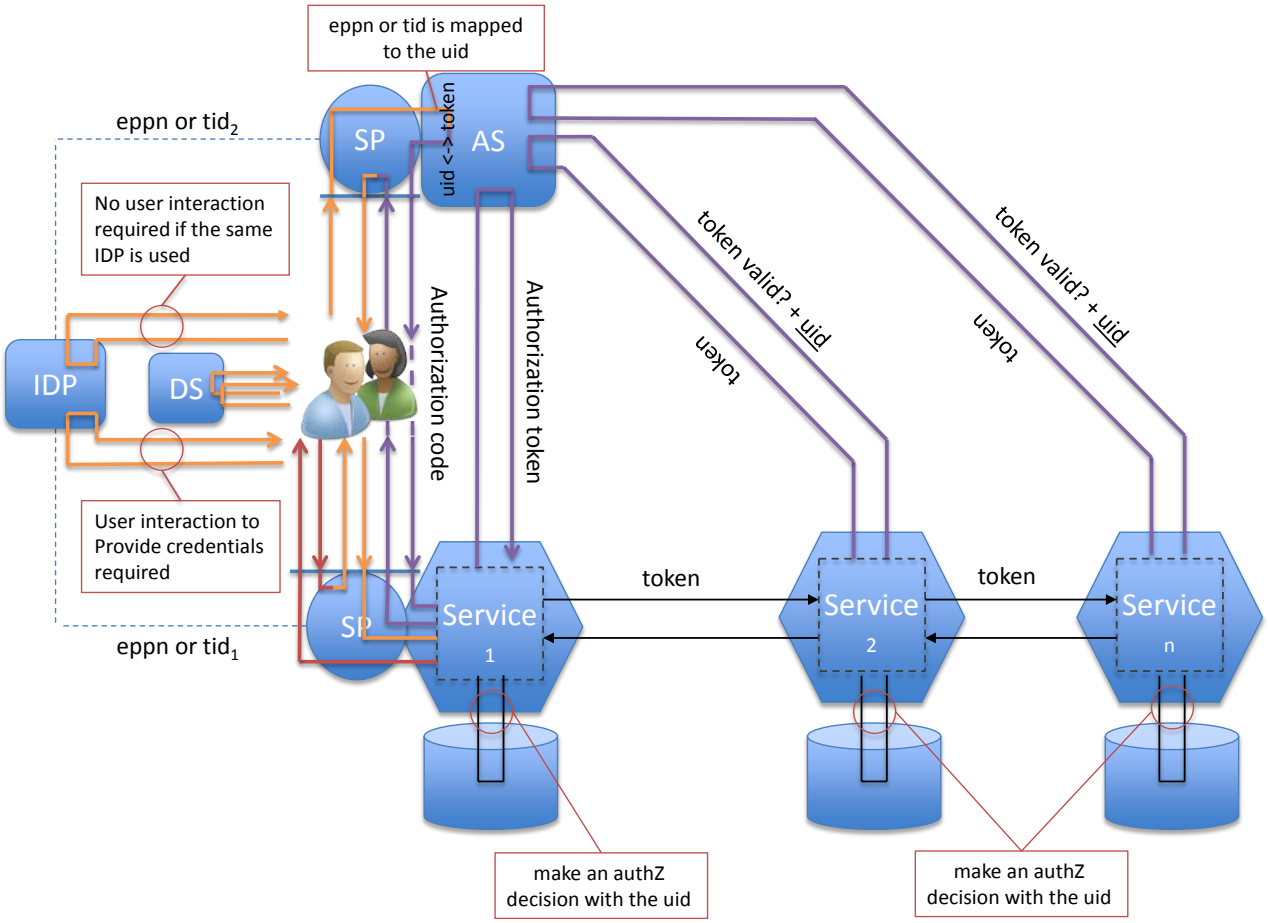


Figure 1. OAuth 2 delegation workflow

Furthermore when later configuration and stability became an issue, the advised WSGI embedding was no longer usable. This was resolved by letting the Apache web server run as a (reverse) proxy in front of an independently running *ndg_oauth* AS. However, the *ndg_oauth* documentation does not cover this, so investigations into the source code were required to achieve this. Documentation covering this setup and the required patches can now be found in the GitHub repository of The Language Archive.[17]

The *ndg_oauth* module is not the only implementation of an OAuth 2 AS. One could, for example, switch to the SURFnet OAuth-Apis AS.[18] The upcoming Section 4.3 reports on some first experiments using this alternative AS.

The solution based on X.509 certificates was not further implemented, but Van Engen and Sallé (2013) state that a smooth transition from OAuth 2 tokens acquired from an AS to certificates acquired from an online CA is possible.

4 CLARIN Use Cases

This section describes a number of cases from the CLARIN infrastructure where integration of services could be extended by means of user delegation. A number of these use cases have already been implemented at a proof-of-concept level. Where applicable, implementation strategies, encountered issues and future perspectives are described.

4.1 CMD Component Registry and ISOcat

This first use case was selected as a pilot because of the availability of development resources within a single institute (the Max Planck Institute for Psycholinguistics) and because the underlying technology

[17] https://github.com/TheLanguageArchive/ndg_oauth
[18] https://github.com/OpenConextApps/apis

stacks of the adapted software components, matches that of the implementation example worked out by Van Engen and Sallé (2013) to a reasonable degree, in particular the client application, which makes use of the Spring framework. Also, the delegation step in this particular use case reflected functionality with (at time of implementation) the potential of real-world application in the production environment.

The Component Registry is part of the Component Metadata (CMD) Infrastructure (Broeder, et al., 2010) implemented by CLARIN. It provides an online editor to metadata modellers to create CMD profiles and components. To enable semantic interoperability, these CMD profiles or components contain references to concept registries. While this use case was developed a prominent registry was the ISOcat Data Category Registry.[19] Within CLARIN, ISOcat has been succeeded by the CLARIN Concept Registry.[20] However, for this paper the experiences to implement the user delegation scenario between the Component Registry and ISOcat are still relevant. The CMD Component Registry editor allowed searching in ISOcat, where the search was initiated by the Component Registry backend, i.e., the backend plays the role of Service 1 and ISOcat that of Service 2 (see Figure 1). Without user delegation only a search for public data categories was possible. Hence the use case is to extend the search for private data categories in the ISOcat users workspace.

To enable this, the Component Registry has been extended with OAuth for Spring Security, providing the following functionality:

1) A method to check if a security token is available in the current session;
2) A method to initiate the request for a security token, i.e., to interact with the *ndg_oauth* AS including logging in and giving permission for delegation;
3) A method to query ISOcat while passing on the security token.

Enabling OAuth for Spring Security required the already present Shibboleth authentication layer to be 'bridged' with Spring Security. This was solved by a simple, though not entirely obvious mapping, involving a custom 'pre-authentication filter' and a dummy 'UserDetailsService'.

On the ISOcat side OAuth for Spring Security could not be used as its implementation is not based on servlet technology. However, this part of the AS interaction is relatively simple. The security token is retrieved from the HTTP header and passed on in a simple check token request to the AS. If the token is valid the identity of the researcher is returned and ISOcat can extend the search to include her workspace.

One implementation issue which still needs to be resolved is the Component Registry's use of frames for the AS interaction. It was pointed out that this hides the URL of the AS and IdP, which makes it hard for the researcher to determine to whom she is providing her credentials.

4.2 CLASS: Cologne Language Archive Services

The CLASS web application[21] implements tools for searching and analysis based on the Poio API,[22] and also provides easy-to-use web interfaces to facilitate field linguists' research. Apart from hosting scripts the main function of the CLASS application is to serve as a gateway to the archives that maintain annotated corpora. The aim is to offer a convenient web-based workflow, which enables the user of the application to access resource files for analysis directly from the repository.

The Cologne use case targets the DoBeS corpus, a core resource hosted by The Language Archive (TLA)[23] at the Max Planck Institute for Psycholinguistics (MPI), a CLARIN center. Most of the collections within the corpus are protected on a personalized level for privacy and ethical reasons. They may only be accessed by the corresponding owner or research group, hence the retrieval of data by external services was unviable in the past. It was soon noticed that this was another case that called for a solution of the delegation issue with the CLASS web application playing the role of S1 and a TLA

[19] http://www.isocat.org/
[20] https://openskos.meertens.knaw.nl/ccr/browser/
[21] http://class.uni-koeln.de/. The CLASS web application was realized as part of the CLARIN-D Curation Projects of Working Group 3, http://de.clarin.eu/en/discipline-specific-working-groups/wg-3-linguistic-fieldwork-anthropology-language - typology/curation-project-1.html.
[22] http://www.poio.eu/
[23] http://tla.mpi.nl/

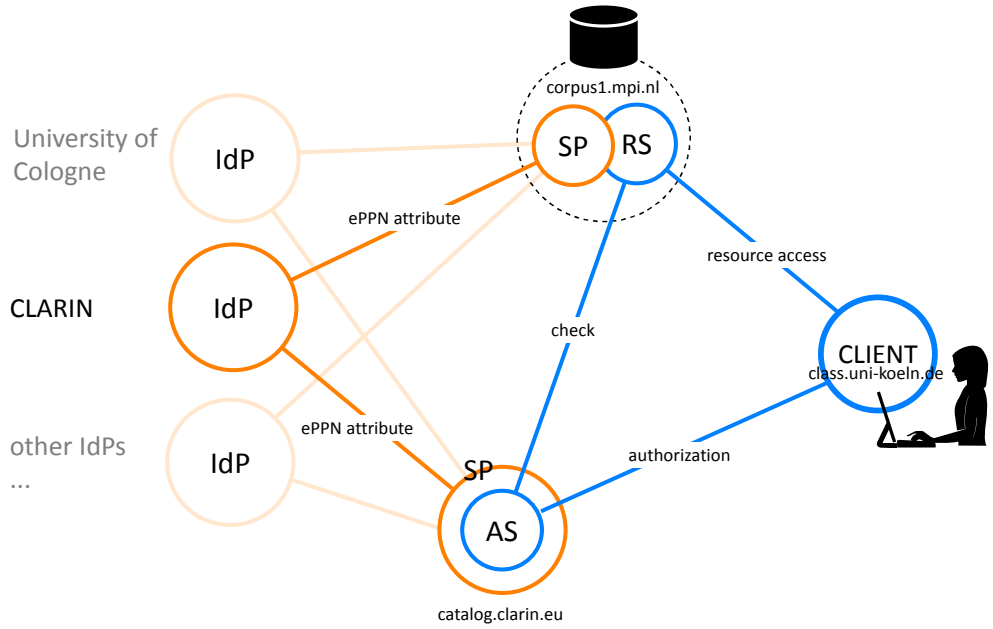


Figure 2. User delegation in the CLASS use case

service that of S2. With the availability of the AS the realization of this layout was possible (see Figure 2).

TLA has implemented a servlet, also known as the TLA Facade Service, which allows delegated access to the resources in the archive. Contrary to ISOcat this servlet can and does use the OAuth for Spring Security. The services provided by the TLA facade are:

1) *accessRights*: receive the access rights (none, read or read/write) the logged-in researcher has for one or more resources;
2) *accessFile*: fetch a specific resource for the logged-in researcher (if she has the right to do so).

The CLASS application uses the *rauth* library[24] written in Python as an OAuth 2 client to talk with the AS and call the TLA facade services. OAuth 2 is specifically designed to reduce complexity on the client side. Tie-ins with common web frameworks are smooth and well documented. Now researchers can run the tools provided by CLASS on resources residing in The Language Archive.

4.3 CLARIN-D ownCloud workspaces

WebLicht[25] is an execution environment for natural language processing pipelines, implemented in CLARIN-D. The online application allows users to construct and execute customized tool chains for text analysis, and subsequently visualize the resulting annotations. OwnCloud[26] is an open-source software system used for file hosting, which provides many features for data sharing and user collaboration. It serves to provide user workspaces, and is deployed and administered at the Forschungszentrum Jülich GmbH (FZJ), a CLARIN-D data center. Currently, in order to save WebLicht results to ownCloud, users must first download the results from WebLicht and then upload to their ownCloud workspace. In this use case, we want to enable users to bypass the download step and directly save results from WebLicht to ownCloud via WebDAV. Both WebLicht and ownCloud are protected by Shibboleth, but behind separate SP's. This scenario exactly demonstrates a user delegation scenario shown in Figure 1, where WebLicht plays the role of Service 1 and ownCloud that of Service 2. This section describes the current state of implementation and further experiments which have been carried out so far.

[24] http://rauth.readthedocs.org
[25] http://weblicht.sfs.uni-tuebingen.de/weblichtwiki/
[26] https://owncloud.org/

The CLARIN-D production installation of ownCloud is protected by an SP through a third party plugin called *user_shibboleth*.[27] Some adjustments to the plugin were made by FZJ to make it function with the ownCloud version currently deployed. In the modified version, the IdP is required to release the Persistent-ID and EPPN attributes. The revised ownCloud plugin maps a hashed version of the user's Persistent-ID to an ownCloud user, and the user's shown name is derived from their EPPN.

An environment that mirrors the actual configuration has been created in order to test implementations and perform experiments using various component options. The remainder of this section reports on the work that was done in this test environment.

The first step taken was to adapt the *user_shibboleth* plugin to allow use behind a reverse proxy and to configure the WebLicht SP to pass the HTTP headers to ownCloud. The patches to the plugin can be found on GitHub.[28]

The next step is to add an extra access point to ownCloud to enable it to process requests with valid OAuth 2 access tokens. See Figure 1, where ownCloud acts as a resource server (Service 2). In order to allow access from WebLicht on behalf of a user, the access point must be exposed outside the SP. Only one official plugin for ownCloud is available which offers this functionality - *user_oauth*,[29] and it is not compatible with the deployed version of ownCloud. Furthermore, it relies on several deprecated third party libraries. The CLARIN-D center in Tübingen addressed and solved the problems with the plugin by essentially reimplementing it.[30]

Next, a server (the AS component in Figure 1) is required which:

1) is capable of authenticating users through a Shibboleth IdP
2) supports token introspection compatible with the *user_oauth* implementation, which was done according to a draft specification (Richer, 2013)[31]

Several options are available for the AS component:

- ndg_oauth AS (as described in Section 3)
- php-oauth-as[32]
- SURFnet OAuth-Apis[33]

Since none of the options fulfill all of the requirements out-of-the-box, each one needs to be assessed individually. *ndg_oauth* AS is capable of authenticating users through a Shibboleth IdP, but it is not compatible with user_oauth and the documentation is fairly sparse. *php-oauth-as* seems to be compatible with *user_oauth* and is being actively developed, but its ability to authenticate users via SAML IdP still remains to be investigated. SURFnet *OAuth-Apis* can authenticate users through a Shibboleth IdP, and can be made compatible with *user_oauth* with only minor changes, thanks to its flexible architecture.

SURFnet *OAuth-Apis* was chosen to be evaluated first for various reasons. It is a Spring application fully compatible with the v2-31 version of the OAuth 2 specification. It provides pluggable authentication and user consent handling, which makes customization very easy. This is particularly important because a specification for token introspection has not yet been finalized and customization will be necessary as the specification evolves. Additional advantages are that it has the most extensive documentation and demo applications, is being actively developed, and has a large user community. A demo has been setup using *OAuth-Apis*. In the demo, a client application namely Testlicht[34] is able to access files on ownCloud.

An alternative to adapting the server to meet the requirements of *user_oauth* would be to implement OpenID-Connect[35] on both the server side and *user_oauth* side. OpenID-Connect is in a sense a layer on top of OAuth 2 providing standardized ways to obtain information about the identity behind an

[27] https://github.com/AndreasErgenzinger/user_shibboleth
[28] https://github.com/weblicht/user_shibboleth
[29] https://github.com/owncloud/apps/tree/master/user_oauth
[30] https://github.com/weblicht/user_oauth
[31] http://www.ietf.org/archive/id/draft-richer-oauth-introspection-06.txt
[32] https://github.com/fkooman/php-oauth-as
[33] https://github.com/OAuth-Apis/apis
[34] https://weblicht.sfs.uni-tuebingen.de/testlicht
[35] http://openid.net/specs/openid-connect-core-1_0.html

OAuth 2 token, and it also provides means to restrict the attribute release. Exploring this promising option is left as future work.

4.4 Virtual Collection Registry

The Virtual Collection Registry (Broeder, Van Uytvanck, & Wittenburg, 2010) is an online service developed within CLARIN that allows users to create collections of resources (including metadata documents) from any location and register them in the CLARIN metadata infrastructure. The service assigns a persistent identifier to the collection upon publication so that it can be referenced as a unit.

A stable version of the Virtual Collection Registry (VCR)[46] is currently available. It has a web front end through which users can log in via Shibboleth to create new virtual collections, edit a collection's metadata and existing resource items, or add new items to a collection through a series of forms. In addition, the service exposes a REST service that supports the creation, manipulation and deletion of collections and resource items. It uses the same authentication policy and methods as the web front end, and therefore the potential for usage in other applications is currently limited.

The addition of support for user delegation to the VCR would allow various other applications to be extended with options to add resources, presented in the context of these applications, to one of the user's own collections, or to create a new collection in the user's workspace within the VCR based on a set of resources. An example of such an application is the faceted browser of the Virtual Language Observatory (VLO),[47] in which users can search for metadata records and associated resources. The connection between the VLO and the VCR could consist of an 'add to collection' option available to the user once search results are shown. When the user chooses this option in this scenario, the VLO connects to the VCR's REST service and request the list of the collection that the user has permissions to work on. After selection of a collection, or alternatively the option to create a new collection, the VLO sends the appropriate request including a list of the selected records to the VCR, which in turn applies the requested changes inside the user's workspace. Repositories at CLARIN centres or elsewhere could provide similar options in their repository search and exploration tools. Examples of such tools would be the hierarchical archive browser[48] of The Language Archive or the search engine of the HathiTrust's digital library.[49]

As the VCR REST service is based on the Java servlet and JAX-RS technologies, it is similar to the TLA facade service described above with respect to adding support for authentication through OAuth 2. Notice that this use case is strictly hypothetical and no efforts towards implementing the described support in either the VCR or the VLO have been taken thus far.

5 Future Work and Conclusion

Apart from these first use cases other uses are possible. For example, in addition to accessing archived resources, CLASS tools could also issue delegated calls to protected remote tools, i.e., web services residing on different sites. The same could be done for WebLicht.

Another potential extension is multi-step delegation: the current solution supports single step delegation, i.e., from S1 to S2, but S2 cannot request a security token from the AS to call a next service, S*n*. Support for such multi-step delegation is currently under investigation. The important question to ask here is how S2 could obtain a new token on behalf of the original user. Perhaps S2 should be able to use the original token to authenticate and get a new token. In order to encode the different authorizations involved in this original token, it will be necessary to implement this in the context of OpenID Connect, perhaps as an extension to it. OpenID Connect adds the necessary handles for the required level of fine-grained attribute release. We are not aware of any (full) solution using OpenID Connect for this type of multi-step delegation.

Not all IdPs release sufficient information for the AS to allow identification of the logged-in researcher. Rather than a universally identical user identifier, such as EPPN (*EduPersonPrincipalName*), the IdP might release a EPTID (*EduPersonTemporaryId*). Although the IdP gives the same EPTID each time the researcher accesses a certain SP (so it can use it to identify the return of the researcher),

[46] http://clarin.eu/vcr
[47] http://clarin.eu/vlo
[48] https://tla.mpi.nl/tools/tla-tools/asv/
[49] http://babel.hathitrust.org (an example selection is already available in the VCR at http://hdl.handle.net/11372/VC-1002)

it gives a different EPTID for the same researcher to each different SP. When the AS and S2 thus are hosted at different SPs the EPTID cannot always be used to identify the researcher. Thus researchers with such an IdP are likely to have problems using delegation.

The *ndg_oauth* AS is currently an experimental service at TLA. In the future this or another AS could be a CLARIN service, but to realize this service, the stability and high availability options have to be investigated first. In this respect the experiments in Tübingen with other AS implementations are very relevant.

The developments within the EUDAT project, especially the B2ACCESS service based on UNITY, are a promising development not directly tackling the delegation issue, but offering flexibility in supporting different technologies that have the potential to provide a solution for the delegation problem. Therefore we consider this a valuable solution to look into. As a first step the OAuth 2 based delegation should be integrated and as a second step support for X.509 delegation can be investigated.

As showcased by the various use cases discussed in this paper support for user delegation is a valuable extension of the CLARIN infrastructure, which will allow further and more fluent integration of key infrastructure components. The experiments to implement these use cases have already helped to make the technology more mature and will in the future continue to do so. A production ready implementation will certainly support CLARIN's mission to enable easy access to language resources, services and tools to the community of humanities scholars.

Acknowledgements

The authors like to acknowledge the valuable support of all the experts in the Big Grid/CLARIN project and the current CSNE informal expert group, especially Willem van Engen who implemented an easy to use and understandable *ndg_oath* and OAuth for Spring Security demo setup.[50] We also like to thank Shakila Shayan for the implementation of the TLA facade servlet.

References

Broeder, D., Kemps-Snijders, M., Van Uytvanck, D., Windhouwer, M., Withers, P., Wittenburg, P., et al. (2010). A Data Category Registry- and Component-based Metadata Framework. Seventh International Conference on Language Resources and Evaluation. Malta: ELRA.

Broeder, D., Van Uytvanck, D., Wittenburg, P. (Eds.). (2010). Language Resource and Technology Registry Infrastructure (CLARIN Report D2R-5b). Retrieved March 18, 2015 from CLARIN: http://hdl.handle.net/1839/00-DOCS.CLARIN.EU-35

Cantor, S. (Ed.) (2012, May). SAML Version 2.0 Errata 05. Retrieved March 18, 2015 from OASIS: http://docs.oasis-open.org/security/saml/v2.0/sstc-saml-approved-errata-2.0.html

Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., & Polk, W. (2008, May). Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Retrieved June 18, 2014 from Network Working Group: http://tools.ietf.org/html/rfc5280

Hammer-Lahav, E. (2010, April). The OAuth 1.0 Protocol. Retrieved June 18, 2014 from Internet Engineering Task Force (IETF): http://tools.ietf.org/html/rfc5849

Hardt, D. (2012, October). The OAuth 2.0 Authorization Framework. Retrieved September 10, 2014 from Internet Engineering Task Force (IETF): http://tools.ietf.org/html/rfc6749

Richer, J. (2013, May 1). OAuth Token Introspection. Retrieved June 17, 2014 from Internet Engineering Task Force (IETF): http://tools.ietf.org/html/draft-richer-oauth-introspection-04

SAML V2.0 Contributors. (2005). Enhanced Client or Proxy (ECP) Profile. In J. Hughes, S. Cantor, J. Hodges, F. Hirsch, P. Mishra, R. Philpott, et al., Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0 (pp. 21 - 31). OASIS.

Van Engen, W., & Sallé, M. (2011). User Delegation in the CLARIN Metadata Infrastructure: connecting the component registry and ISO-DCR - Part I - Research. CLARIN/BiG Grid. Retrieved March 18, 2015 from NIKHEF: http://wiki.nikhef.nl/grid/images/6/66/Clarin-security_for_web_services-research-report010.pdf

[50] https://github.com/wvengen/oauth2-demo

Van Engen, W., & Sallé, M. (2013). User Delegation in the CLARIN Metadata Infrastructure: connecting the component registry and ISO-DCR - Part II - Implementation. CLARIN/BiG Grid. Retrieved March 18, 2015 from NIKHEF: http://wiki.nikhef.nl/grid/images/1/17/Clarin-security_for_web_services_inplementation.pdf