Access Control in Linked Data Using WebID

A Practical Approach Validated in a Lifelong Learning Use Case

Pascal Mainini
Bern University of Applied Sciences (BFH)
Institute for ICT Based Management
Höheweg 80, CH-2502 Biel/Bienne
pascal.mainini@bfh.ch

Prof. Dr. Annett Laube-Rosenpflanzer
Bern University of Applied Sciences (BFH)
Institute for ICT Based Management
Höheweg 80, CH-2502 Biel/Bienne
annett.laube@bfh.ch

ABSTRACT

Linked Data technologies become increasingly important in many domains. Key factors for their breakthrough are *security and trust*. Classical means for access control lack granularity when *parts* of the Linked Data graph must be protected. The WebID, combining semantic web concepts with methods from certificate based authentication and authorization, seems promising to fulfill all requirements concerning security and trust in the semantic web.

In the PerSemID project, we challenged the WebID technology with a *fully implemented proof-of-concept (PoC)* addressing a workflow coming from the domain of lifelong learning and student mobility. In our use case of study enrollment, we used WebIDs for authentication and to grant access to parts of triple stores, during cross domain triple store interactions to exchange data between stakeholders.

CCS Concepts

•Security and privacy → Authentication; Access control; Authorization; Usability in security and privacy;

Keywords

Semantic Web, WebID, Linked Data, Access Control, Authentication, Authorization

1. INTRODUCTION

PerSemID[1], the successor of the CV3.0 project[2], investigated issues remaining open in practical applications of the WebID[3] technology: its use for authentication and authorization in triples stores.

While not questioning WebID's general security properties – they are implied by the underlying mechanisms based on client certificate authentication given by Transport Layer Security (TLS) – we investigated the question of trust, or more specifically the question of *level of assurance (LoA)*[4] in WebIDs. The LoA, an important concept in identity and access management, states a quality level regarding authentications.

The second aspect concerns the application of WebID for access control to resources, operated by independent parties and in distributed environments. Here, we focused on triple stores and platforms for document management.

Our use case in the domain of lifelong learning and student mobility shows the use of Linked Data for administrative processes in enrollment for studies. The concepts developed in our complete proof-of-concept prototype can easily be adapted to similar processes in other domains.

SEMANTICS 2016: Posters and Demos Track
September 13-14, 2016, Leipzig, Germany

2. RELATED WORK

PerSemID lies at the intersection of two domains: identity and access management (IAM) and semantic web based technologies with a focus on attribute transfer and document management.

Web Access Control[5] is one of the first approaches in providing authorization based on WebIDs acting on the level of HTTP resources.

Universal Access Control (UAC)[6], which we used in CV3.0, goes further and provides access control at the level of individual triples. Like UAC, the Privacy Preference Ontology[7] provides access control at triple level.

WebID+ACO[8] is an ontology for authorization which focuses on adding a role-based authorization model to HTTP.

S4AC[9] is a vocabulary for creating access control policies focusing on named graphs. S4AC is used by the SHI3LD project[10] for specifying permissions.

The MyProfile project offered an IDP service for WebID as well as a platform for social networking. Online resources of MyProfile are not available anymore, detailed information can be found in Sambra[11]. Recently, a new initiative called Solid[12] seems to take up on MyProfile.

3. WEBID

WebID authentication builds on the authentication of a client using X.509[13] client certificates. Functionality for using such certificates is present in all major browsers.

To deliver additional information (for example personal attributes) and to establish URIs as identificator for a particular entity, WebID references a FOAF profile[14] using a standard extension of X.509 (the *Subject Alternate Name (SAN)* field). Figure 1 gives an overview of authentication using WebID. The client (identified by its X.509 client certificate with corresponding key pair) wants to authenticate to an application running on a (web-)server. The webserver retrieves the FOAF profile referenced in the SAN of the certificate and compares the information about the public key against the information obtained from the client certificate in the TLS handshake. If they match, authentication is successful. If desired, additional potentially signed attributes and other information can be retrieved from the profile.

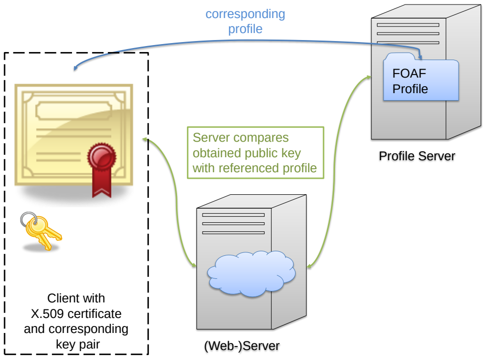

corresponding profile
FOAF Profile
Profile Server
Server compares obtained public key with referenced profile
Client with X.509 certificate and corresponding key pair
(Web-)Server

Figure 1: WebID working principle

Anyone can issue a self-declared WebID by simply generating an appropriate certificate and publishing a corresponding FOAF profile document on a webserver.

The search for trust in WebID is a general problem which can also be found in other systems that use public key cryptography – as for instance classical X.509 certificates for websites or secure mail. Today, still the most common approach in creating trust is the hierarchical Public Key Infrastructure (PKI) model with certificate authorities.

SuisseID, a PKI operated according to national signature laws[15] by privately held certificate authorities (accredited by the state), provides X.509 certificates for authentication and digital signing on hardware tokens. Besides the PKI, SuisseID also runs an attribute authority which provides additional information about the holder of a certificate, like name, date of birth or gender.

Being widely recognised, accepted, and having a very high level of trust, SuisseID would be an ideal partner for strengthening the LoA of a WebID. Furthermore, the attribute authority functionality could seamlessly be integrated into the FOAF profile server, thus providing the same attributes with the same level of assurance for the Linked Data world.

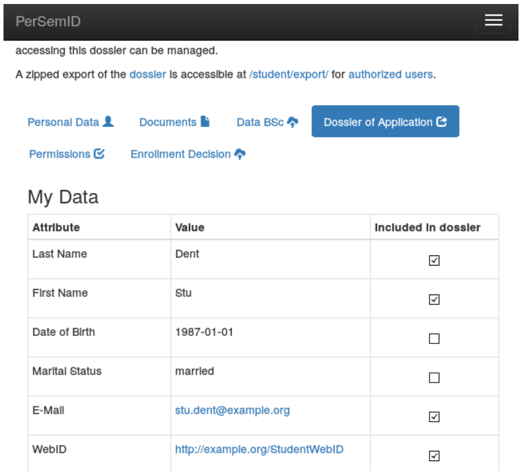
Extending SuisseID with WebID is technically not a hard problem: Certificates issued by SuisseID must be extended to include the proper SAN extension, containing the URI to the corresponding FOAF profile, and the issuing certificate authority must operate a webserver for serving these FOAF profile documents accordingly.

We took on the integration approach as described and validated it prototypically using the demo SuisseID identity provider from [16] (details are in [17]).

Even though not being a new technology (surfaced end of 2008), WebID has not found broad adoption so far. While looking simple and flexible at first sight, it suffers from some issues which have been noted by others [18] already. Most notably, the overall user experience of WebID seriously hinders broad adoption of the technology and unfortunately, there seems to be no intention on the part of browser vendors to change this anytime soon. Additionally, nowadays the user typically owns a multitude of different devices, making certificate management nearly impossible.

4. USE CASE

Our PoC aimed to challenge the application of WebID in a working implementation of a real-life scenario. The workflow conducted in the enrollment for master studies was chosen as our exemplary use case. This workflow involves three primary actors: A *student* who has successfully obtained a bachelor's degree (and may have additional qualifications), the institution at which this degree has been obtained (called *bachelor university*) and finally the institution at which the student whishes to enroll for master studies (the *master university*).



accessing this dossier can be managed.

A zipped export of the dossier is accessible at /student/export/ for authorized users.

My Data

Attribute	Value	Included in dossier
Last Name	Dent	☑
First Name	Stu	☑
Date of Birth	1987-01-01	□
Marital Status	married	□
E-Mail	stu.dent@example.org	☑
WebID	http://example.org/StudentWebID	☑

Figure 2: Configuration of dossier of application by the student

A fully working prototype has been made available [19]. We also produced a screencast [20] demonstrating the main workflow between all involved parties

From a technical perspective, PerSemID builds upon the concepts of a personal, semantic curriculum vitae developed in CV3.0. The architecture for a corresponding platform for serving and maintaining such a CV has been defined in CV3.0's *Content Access Service (CAS)*[21]. The CAS is a RDF triple store with additional document management capabilities as well as an access control layer.

4.1 Actors and Their Actions

In a first step, the student prepares a *dossier of application* which contains all relevant information about the degree obtained and possible additional data in form of documents. Provenance of this data is either personal information entered by the student directly or data obtained from the bachelor university in the *bachelor dossier*. The bachelor dossier is issued by the bachelor university as a single file, containing Linked Data about the degree obtained and optionally additional documents. All this data is then stored in the student's CAS and the student can freely choose to include/exclude data per application at a master university (see Figure 2).

After having created the dossier of application, the student authorizes the master university to access the dossier by restricting access based on the university's WebID which is assumed to be publicly available.

Next, the addressed master university picks up the dossier by accessing it on the student's CAS. Following a review of all the material in the dossier, a decision regarding acceptance to master studies can be made. Now, the master university in turn stores its decision in its CAS and authorizes the WebID, given by the student, to access it.

In the last step, the student finally retrieves the decision from the master university.

4.2 Architecture and Implementation

4.2.1 *Content Access Service*

There is no ready-made product similar to a content access service as specified by [21], thus the needed functionality had to be implemented in the PoC itself. A large range of (mature) triple stores is available (see in [22]). We chose to use Apache Jena[23], that supports SPARQL 1.1 update together with other requirements.

A deliberately reduced set of document management capabilities has been implemented in the PoC code itself.

The CAS serves as storage for all metadata related to each actor and also as location for all application-specific configuration data, like file system paths or granted permissions. An example for the contents of the student's graph, including a granted permission for the WebID `hmsc.example.org` can be seen in Listing 1.

Listing 1 Partial example data of a student

```
@base <http://example.org/Student> .
@prefix rdfs: <http://www.w3.org/2000/01/
    rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/
    XMLSchema#> .
@prefix s: <http://persemid.bfh.ch/vocab/
    student#> .

<#> a s:Student ;
    s:webid <http://example.org/
        StudentWebID> ;
    s:name "Dent" ;
    s:vorname "Stu" ;
    s:email "stu.dent@example.org" ;
    s:matrikelnummer "1-234-56" ;
    s:permission <http://hmsc.example.org/
        webid#id> .
```

Documents, which can be uploaded by the student and the bachelor university, are given a unique ID and stored on the file system. Metadata needed by the server for interacting with them is again stored in the triple store, in the named graph of the respective actor.

4.2.2 *Server Application*

The whole server application has been written in JavaScript and is running on node.js[24]. HTTP-functionality has been realized using the widely deployed middleware layer connect[25] which makes creation of applications serving a variety of different requests straightforward. All communication between the frontend application and the server runs over a single HTTPS port.

On both sides, client and server, we make use of JavaScript RDF libraries like rdf-ext[26] and ld2h[27]. An overview of the architecture is given in Figure 3.

4.2.3 *WebID Identity Provider*

All functionality needed for WebID authentication has also directly been implemented in the PoC itself, based on our experience in the implementation of the WebIDP[28] application, an identity provider for WebID developed in CV3.0. A dedicated URL of the webserver serves the FOAF profiles referenced by the client certificates. The certificates for all actors were generated directly using OpenSSL with respective configuration files.

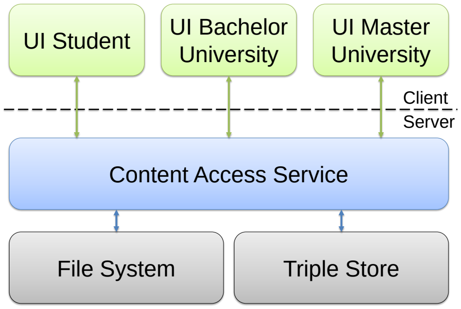


Figure 3: Overview of architecture

4.2.4 *Cross-Domain Triple Store Interaction*

As described in Section 4.1, all actors follow a defined scheme of interaction. In this scheme, there are three data exchanges: download of bachelor data by the student from the bachelor university, download of data from the student by the master university and finally, download of data from the master university by the student.

This can be generalized as a concept for sharing data between triplestores or *cross-domain triple store interaction*. Multiple methods for implementing such interactions could be thought of, we considered the following three: (1) cross-site sharing using HTTP access control, known as *CORS*[29]; (2) proxying of data on the server side; and (3) explicitly channeling data through the client.

Being limited by the same-origin policy, that restricts how a document or script loaded from one origin can interact with a resource from another origin, a direct interaction between the client-side program logic and the content access service of the remote party in an exchange cannot be implemented – even considering the fact, that in our PoC scenario, all content was served from the same server. This problem could be circumvented with HTTP access control (CORS), which allows for a relaxation of the restrictions imposed on the client. By doing so, we would face another problem: in order to be able to dynamically adjust the needed HTTP headers, parties exchanging data would have to know each other in advance – rather an unlikely situation in a real world scenario.

One notices, that this is shifting control towards the server, leading to another option where the server acts as a proxy for the data to be exchanged. Being a seemingly straightforward approach, this method has some serious drawbacks as well. We would have strong concerns regarding security if the server could be instructed by the client to act as an open proxy interacting with unknown destinations. Also, for the purpose of our PoC, hiding the exchanges between actors is not optimal for the demonstration of the implemented functionality.

So we finally set with the third option and implemented a very explicit data exchange using ZIP-files which are downloaded by an actor from the remote party and manually imported into their own CAS. While this may look odd or even ancient at a first glance, it has some great benefits for our validation work, which amongst others are: (a) explicit WebID authentication and authorization are possible – our main objective in this case; (b) separation of the actors and adminstrative borders are clearly visible; and (c) interaction with files is well known to the user.

5. CONCLUSIONS

Our prototype showing the process of study enrollment demonstrated, that by using Linked Data technologies, concrete and practical administrative workflows can be implemented easily and without hassles. Authentication and authorization using WebID stands the test regarding security requirements in that area – an integration into other, trusted identification systems such as SuisseID is technically possible and would enhance the WebID in terms of trust.

The prototype gave us insights in cross-domain triple store interaction and provided a model for future implementations of processes and workflows based on Linked Data technologies. During the implementation, we encountered some issues, most notably related to the same origin policy of modern browsers. For these issues, we gave an overview of possible solutions and described the one chosen.

Besides technical problems, our research clearly showed weak points in WebID. Regarding broader acceptance of the technology, as means for authentication and especially as a "token" for permission handling, future work for better integration, portability and especially userfriendlyness must be undertaken. Here, we are particularly interested in approaches taken by recent projects like Solid – and whether those will be successful in solving these issues.

6. REFERENCES

[1] PerSemID project homepage. Online, last retrieved April 2016, http://persemid.bfh.ch.

[2] CV3.0 project homepage. Online, last retrieved April 2016, http://cv3.bfh.ch.

[3] Toby Inkster, Henry Story, and Bruno Harbulot. WebID authentication over TLS. Online, last retrieved April 2016, https://www.w3.org/2005/Incubator/webid/spec/tls/.

[4] ISO/IEC 29115:2013: Information technology – security techniques – entity authentication assurance framework, 2013.

[5] Henry Story, Tim Berners-Lee, et al. Web access control. Online, last retrieved April 2016, https://www.w3.org/wiki/WebAccessControl, 2016.

[6] Thomas Bergwinkl. LDApp - a JavaScript linked data stack. CEUR Workshop Proceedings, International Semantic Web Conference 2014. Online, last retrieved April 2016, http://ceur-ws.org/Vol-1268/paper13.pdf.

[7] Owen Sacco and Alexandre Passant. A privacy preference manager for the social semantic web. CEUR Workshop Proceedings Vol-781, SPIM2011. Online, last retrieved April 2016, http://ceur-ws.org/Vol-781/paper6.pdf, 2011.

[8] D. Tomaszuk, H. Gebhardt, and M. Gaedke. WebID+ACO: A distributed identification mechanism for social web. 2011.

[9] S. Villata, N. Delaforge, and F. Gandon. S4AC vocabulary specification 0.2. Online, last retrieved April 2016, http://ns.inria.fr/s4ac/v2/s4ac_v2.html, 2011.

[10] L. Costabello, S. Villata, F. Gandon, and N. Delaforge. SHI3LD: Context-aware authorization for graph stores. Online, last retrieved April 2016, http://wimmics.inria.fr/projects/shi3ld, 2012.

[11] Andrei Vlad Sambra. *Data ownership and interoperability for a decentralized social semantic web.* PhD thesis, Institut National des Télécommunications, 2013.

[12] Andrei Vlad Sambra, Dmitri Zagidulin, Tim Berners-Lee, et al. Solid specification. Online, last retrieved April 2016, https://solid.github.io, 2016.

[13] D. Cooper et al. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Online, last retrieved April 2016, https://www.ietf.org/rfc/rfc5280.txt, 2008.

[14] Dan Brickley and Libby Miller. FOAF Vocabulary Specification 0.99. Online, last retrieved April 2016, http://xmlns.com/foaf/spec, 2014.

[15] SR 943.03: Bundesgesetz über Zertifizierungsdienste im Bereich der elektronischen Signatur. Swiss National Law, https://www.admin.ch/opc/de/classified-compilation/20011277/index.html, 2008.

[16] SuisseID SDK/Java. Online, last retrieved April 2016, https://www.e-service.admin.ch/wiki/display/suisseid/SuisseID_SDK_Java.

[17] Michele Santomauro. *Middleware solutions for e-Government Interoperability Frameworks.* PhD thesis, Università degli studi della Basilicata, 2014.

[18] FOAF-protocols mailinglist thread. Online, last retrieved April 2016, http://markmail.org/message/h7f2aldjeqv3l5dd.

[19] PerSemID, PoC Sourcecode. Online, last retrieved April 2016, https://github.com/mainini/persemid-usecase.

[20] PerSemID, PoC screencast. Online, last retrieved April 2016, http://persemid.bfh.ch/screencast.html.

[21] Pascal Mainini. CV3.0 content access service specification. Online, last retrieved April 2016, http://cv3.bfh.ch/architecture-implementation.html.

[22] Lars Marius Garshol. RDF triple stores — an overview. Online, last retrieved April 2016, http://www.garshol.priv.no/blog/231.html.

[23] Apache Jena triplestore. Online, last retrieved April 2016, https://jena.apache.org.

[24] node.js JavaScript runtime. Online, last retrieved April 2016, https://nodejs.org.

[25] connect node.js middleware. Online, last retrieved April 2016, https://github.com/senchalabs/connect.

[26] rdf-ext RDF interfaces extension. Online, last retrieved April 2016, https://github.com/rdf-ext/rdf-ext.

[27] ld2h, linked data to html. Online, last retrieved April 2016, https://github.com/rdf2h/ld2h.

[28] Pascal Mainini. WebIDP, identity provider for WebID. Online, last retrieved April 2016, http://webidp.bfh.ch.

[29] HTTP access control (CORS). Online, last retrieved April 2016, https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS.