WebID+ACO: A distributed identification mechanism for social web

Dominik Tomaszuk
Institute of Computer Science, University of Bialystok, Poland
dtomaszuk@ii.uwb.edu.pl

Martin Gaedke, Hendrik Gebhardt
Faculty of Computer Science, Chemnitz University of Technology, Germany
{martin.gaedke, hendrik.gebhardt}@informatik.tu-chemnitz.de

Abstract. This paper defines an approach to managing digital identity requiring special-purpose technology on the browser client. We propose a mechanism using standards, such as HTTP(S) extended with WebID Protocol and Semantic Web ontologies and vocabularies. We present a scalable method that allows user authentication and authorization to work across multiple web-sites, enterprises, devices, and browsers in a uniform and easy-to-use manner.

Keywords. Semantic Web, Resource Description Framework (RDF), WebID, ontology, authentication, authorization, access control list, linked data, public key infrastructure, social web

1. Introduction

Identity have been at the center of how we interact with the Web in the last decade. The explosion of social networking sites has brought the world closer together as well as created new points of pain regarding ease of use and the Web.

Current social networks are isolated from each other. They are not designed to use a central source or copy existing data. Because of this behavior the name social network site silos was established. This produces further problems like multiple time consuming maintenance work or even wrong or missing personal data. Also friends and network partners could be affected. The searches for the right data in all of the silos could consume even more time. Because much effort was spent to maintain the data it could be considered as a base for authorization purposes. But the possibilities are limited or don't exist. Another problem is the Social Network lock-in. If you want to leave one, you lose your network. There are, in the majority of cases, no possibilities to leave a network and to keep your data and established friends in a usable manner.

Remembering login details, passwords, and sharing private information across the many websites and social groups that we are a part of has become more difficult and complicated than necessary.

We based our concept on WebID where a user chooses their preferred identity provider according to their preferences and defines access control lists.

In this paper we will first introduce the WebID approach. In the following section we will discuss the opportunities of using social network data as means for authorization descriptions. In Section 3, we introduce an ontology that enables and demonstrate its application for new kinds of authorization scenarios as described above. The paper ends with conclusions.

2. Authentication over WebID

Prior to the authorization, the authentication has to be done. In this Section WebID [1] is presented in the context of authentication.

WebID is an open standard for login and identity. It uses technologies like the Hypertext Transfer Protocol (HTTP) and Transport Layer Security (TLS) [2] as well as Semantic Web vocabularies such as Friend-of-a-Friend (FOAF) [3]. WebID is a replacement for the traditional login with username and password. It uses the certificate exchange mechanism of the TLS, which is used in secure HTTP connections that is supported by almost all modern web browsers. Within the exchange the browser requests the user to select a certificate for the login. The

WebID certificate is annotated with an URI of the WebID profile that uniquely identifies a person, organization, or other user agent.

Contrary to OpenID [4], with WebID you do not have to remember the URI, which identifies your profile because it is stored in the browser or keychain of the system. As a matter of fact, WebID is compatible with OpenID and there are several reasons OpenID will be used instead or complementary of WebID. Many services accept OpenID as authentication method and services like openid4.me allow to login with a WebID in OpenID-enabled websites. For the following authorization it is very important to uniquely identify a logged in user. WebID authentication sequence is presented is Fig. 1.

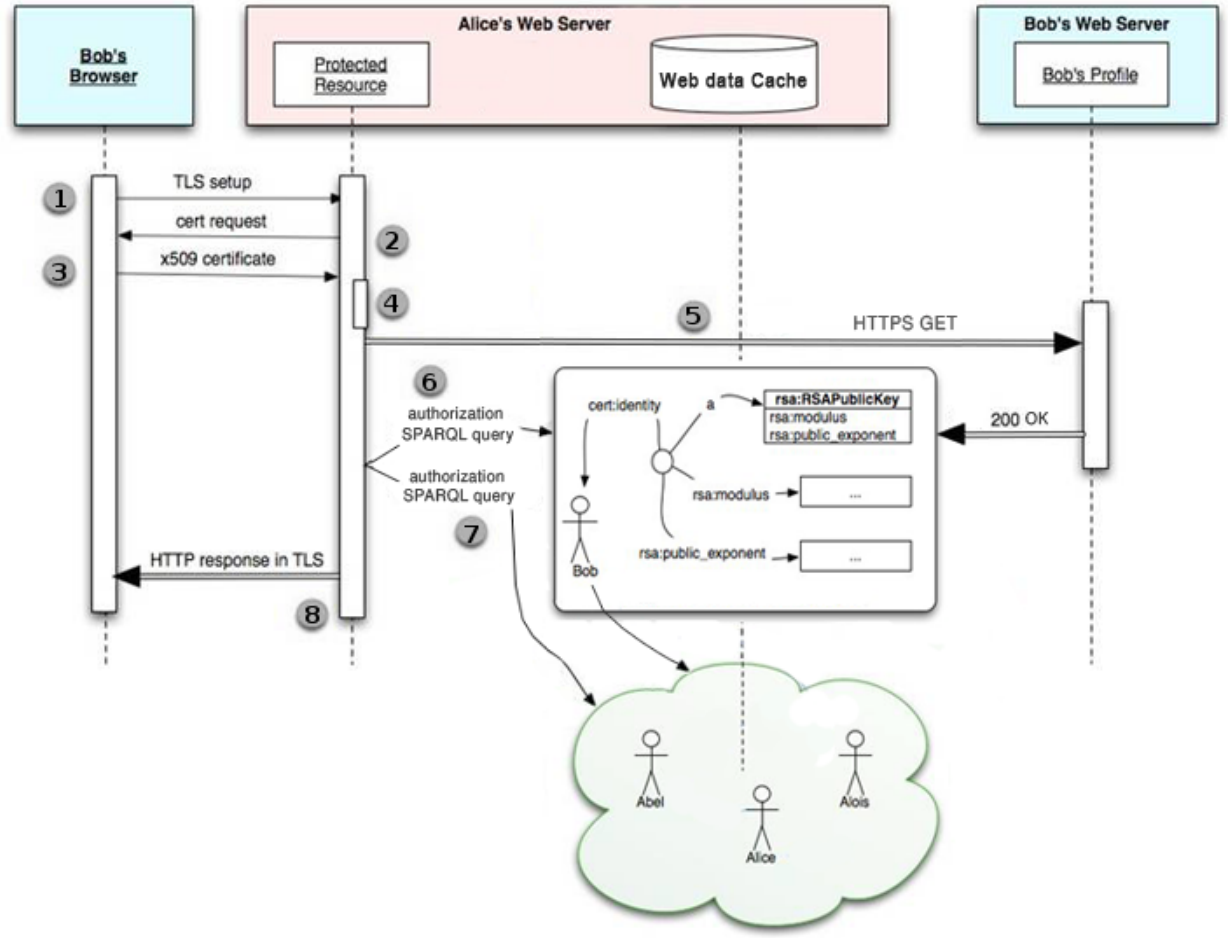

Fig. 1: Authentication sequence.

1. Bob requests Alice's protected HTTPS resource.
2. Alice's web server requests the client certificate on the TLS connection started above.
3. Bob's browser presents him with a selection of identities to choose from. Having selected one, the corresponding X509 certificate is sent to Alice's server. It contains Bob's WebID - https://bob.net/id/bob#me.
4. Alice's server:
    a. checks that Bob's browser is in possession of the private key corresponding to the public key sent in the certificate, as specified by TLS.
    b. extracts the URI from the Subject Alternative Name field of the certificate, which is known as Bob's WebID: a global identifier that refers to Bob via a document that describes him, in a machine readable way using W3C standards.
5. Alice's server fetches Bob's WebID Profile at https://bob.net/id/bob in the example if an up to date version is not in the cache.

6. Alice's server checks via SPARQL query [5] that the profile relates Bob's WebID to the public key found in the certificate. If they match then she knows that she is in communication with the agent referred to by https://bob.net/id/bob#me.
7. Bob's identity is then checked as to its position in a graph of relations in order to determine trust according to some criteria decided by Alice combined with information from the cloud. It is presented in Section 3.
8. Access is granted to read and write, read-only, or denied and a representation is returned [6].

3. Authorization over Access Control Ontology

In this Section the proposed Access Control Ontology is presented (in the sequel denoted by ACO). ACO is an ontology describing roles, their permissions, and allowed or permitted actions on web-site. It allows the description of access control lists for groups of agents. ACO is a descriptive vocabulary expressed in Resource Description Framework (RDF), RDF Schema (RDFS) and Web Ontology Language (OWL). The presented ontology is written in the RDF/XML syntax and Terse RDF Triple Language syntax [7].

We define an authorization a ∈ AuthZ as a tuple of the form < *role; action* > where *role* ∈ *R* and *action* ∈ *ACT* . The Agent class is defined in FOAF [3]. FOAF agents are assigned (*hasRole* property) to roles (Role class). Agents should have a minimum of one role. Roles may have names (*roleName* property). There is also default policy for the role (*DefaultPolicy* class), which could deny (Deny class) or permit (*Permit* class) access to data. It should have exactly one default policy. Roles are assigned (*hasPermission* property) to their permissions (*Permission* class). Let *P* be the permissions, *A* be the agent and *DP* be the default policy, then role *R* ∈ *{a p, dp}* with a ∈ *A*, *p* ∈ *P* and *dp* ∈ *DP*.

The permissions are assigned (*hasAction* property) to actions (*Action* class). The actions *ACT* specify roles which are granted to access the web-site, as well as what operations are allowed or forbidden on the web-site. We propose two types of actions that can be combined with each other. These types of permission are divided into two groups: read (*Read* class), and write (*Write* class).The read permissions allow the execution of the HTTP requests: GET and HEAD. We also suggest in this mode to show all HTML form controls with *readonly* attribute. The write permissions allow the execution of the HTTP requests: POST, PUT and DELETE.

The ontology is presented on Fig. 2.

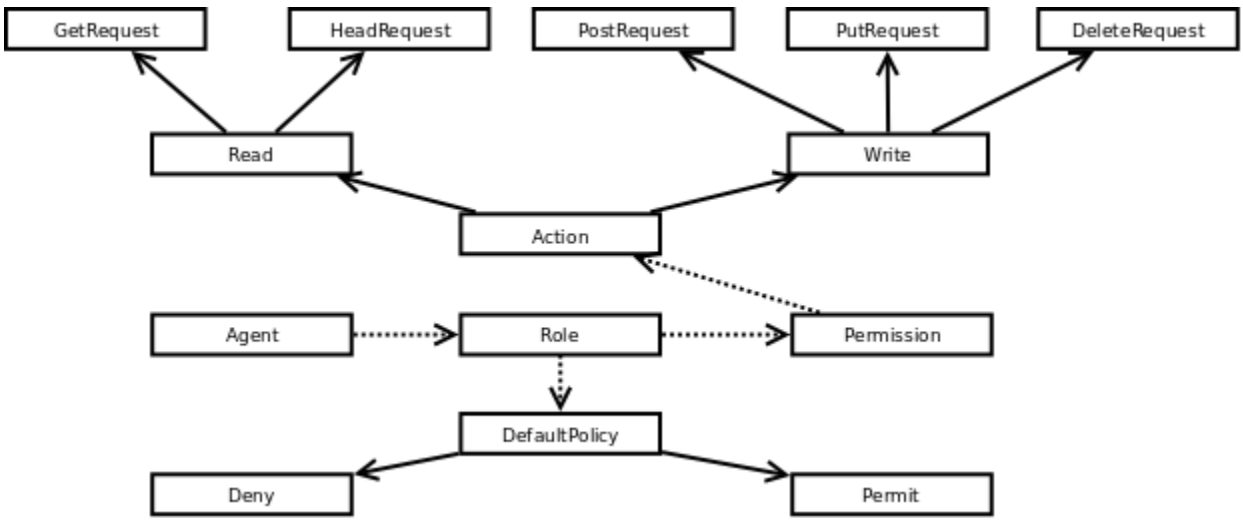

Fig. 2: Access Control Ontology

An example of access control list based on proposed ontology is presented on listing 1.

```
@prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix http: <http://www.w3.org/2006/http#> .
@prefix aco:  <http://example.org/aco#> .
```

```
_:a01 a foaf:Agent ;
      aco:userName <http://example.org/card#me> ;
      aco:hasRole [ a aco:Role ;
                     aco:roleName "friends" ;
                    aco:hasDefaultPolicy aco:Permit ;
                    aco:hasPermission [ a aco:Permission ;
                                        aco:priority "10"^^xsd:int ;
                                        aco:hasAction http:Get ] ] .
```

Listing 1: Example of ACO-based access control list

Current ontologies for access control, like the WebAccessControl [8] allow to define standard rules. Our approach of an Access Control Ontology additionally has roles. It is role-based access control [9]. They will be more flexible than static group definitions. WebAccessControl is, at the moment, a very simple approach to protect information on the web. The ontology supports the following four modes of access: Read, Write, Append and Control. Users are identified via WebID or as a part of a defined group. The group definition can be done externally and used in your Access Control List (ACL) as an URI.

4. Use cases

In this section representative use case of WebID+ACO is described. By allowing users to have more and more of their data hosted on the web applications new challenges have emerged. Users must control, which web applications, that store their resources, must ensure that those resources are protected according to their requirements.

With the increasing amount of resources, Alice decides that it is much better to move all of her resources to specialised web applications. To meet the goals of her undertaking, Alice sets up accounts on storing pictures application and publish there some pictures. Considering this action, Alice decides to move authentication and authorization from her web site to dedicated WebID provider. When an account is set up on an WebID provider, Alice configures each of her web applications to delegate access control decision. She has a friend who name is Bob. Alice enables role named *friends* access to the pictures. She assigns Bob to *friends* role. Alice notifies Bob that she wants share some pictures. She uses for this Semantic Pingback [10]. Her web server check the access control lists to Bob's role via SPARQL query. Then it determines the type of access: Bob can only see and download this pictures. Next, Bob obtains an read-only access Alice's pictures.

A Content Management System (CMS) can be a possible application of an ACO. Several authors, which are identified for instance via WebID, are participants of such a system. Based on their current role, they are able to interact with the system. Lucy is author of an article in this CMS. The article is identified by an URI (http://example.org/a/23). She can select some reviewer for this article, which are able to edit it. After publishing, all users with a valid WebID can comment it. Therefore the namespace for comments (http://example.org/a/23/c) has to be writable for other users. The user are able to edit their own comments and, if they are in the role of a reviewer for the article, able to edit and delete comments of other users. In this use case is shown that access control are important even if the participating users, like a reader of an article who writes a comment, are unknown. The mapping between a comment and the corresponding editor is essential. Depending on these characteristics, semantic web technologies, like Semantic Pingback [10], can be applied.

In enterprise systems the permission system can additionally based on the division of an employee in a company. Then the access control is based on the role and the division of the specific employee. That is a common application flow in companies. A standardized declaration allows reusing these permissions in the whole company, even though the employee varies the division. Within the company, there is no need of a centralized database of profile information. Each employee is able to maintain its personal and project specific information. It depends on each individual settings of the access control who is able to access which information. Additionally there can be a company- or division-wide policy that regulates specific settings and overrides user-defined permissions.

Based on the enterprise approach, a federated access control systems is under normal circumstances very complex. With a distributed approach of an access control it would be relatively simple to implement. The

definition of rules will be done like in any other cases with the exception of several involved companies. If company A, B and C are working together in a project, they only have an initial exchange of the credentials, for instance the WebID URIs of involved employees, and beyond of this the standard access control ontology will be used. An endpoint, which is open for external organizations, is needed to evaluate profile information between several organizations. Especially this case requires a fine-grained access control, since there are internal secrets of a company that should not be leaked out.

5. Conclusions

The problem of how to adjust authentication and authorization has produced many proposals. Most of them are hard to use without requiring special-purpose technology on the web browser, hence making the problem seem difficult.

We have presented a simple and thought-out proposal. We believe that our idea is an interesting approach, because it is web browser independent. We have proposed an identification protocol dedicated to web browsers that is universal and distributed. It uses HTTP, WebID and ontologies: FOAF and ACO. Our proposal can work either with mobile and other devices or a web browser and servers. A crucial advantage of the proposal is that the identification mechanism is distributed and adopts linked data.

Acknowledgements

The authors would like to thank WebID Incubator Group from the World Wide Web Consortium, particularly Henry Story and Thomas Bergwinkl. Professor Henryk Rybinski's comments and support were invaluable.

References

1. S. Corlosquet; et. al. WebID 1.0 Web Identification and Discovery. 10 February 2011. W3C Working Draft. URL: http://www.w3.org/2005/Incubator/webid/spec/
2. E. Rescorla. HTTP Over TLS. May 2000. Internet RFC 2818. URL: http://www.ietf.org/rfc/rfc2818.txt
3. D. Brickley and L. Miller. FOAF Vocabulary Specification 0.98. 9 August 2010. URL: http://xmlns.com/foaf/spec/
4. D. Recordon and D. Reed. OpenID 2.0: a platform for user-centric identity management. Proceedings of the second ACM workshop on Digital identity management. 2006.
5. E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. 15 January 2008. URL: http://www.w3.org/TR/rdf-sparql-query/
6. J. Sayre and H. Story; et. al. The WebID Protocol & Browsers. URL: http://bblfish.net/tmp/2011/04/26/
7. D. Tomaszuk. Access Control Ontology. 2 May 2011. URL: http://ii.uwb.edu.pl/~dtomaszuk/access/
8. Access Control W3C Wiki page uses the http://www.w3.org/ns/auth/acl ontology
9. R.S. Sandhu, E.J. Coyne, H.L. Feinstein and C.E. Youman. Role-based access control models. Computer 1996.
10. S. Auer, P. Frischmuth, S. Tramp and H. Story. Semantic Pingback Vocabulary. 7 September 2010. URL: http://docs.aksw.org/SemanticPingback/namespace.html