How to Publish Privately

Nuno Bettencourt[1], Nuno Silva[1], and João Barroso[2]

[1] GECAD, Instituto Superior de Engenharia do Porto, IPP, Portugal
{nmb,nps}@isep.ipp.pt
[2] INESC TEC and Universidade de Trás-os-Montes e Alto Douro, Portugal
jbarroso@utad.pt

Abstract In a world overwhelmed by constant data creation and manipulation, where privacy is becoming a real concern, topics like data usage control, accountability, provenance, protected sharing of resources and trustworthiness of knowledge sources are becoming main topics of discussion among communities of interest. In this paper enhancements are proposed for an existing framework that tackles some of the afore mentioned issues namely data provenance, usage control and accountability. Such proposals consist of providing means for publishing resources in a private manner hereby making websites behave like meshes of hyperlinked resources from different domains, not only for resources publicly published but also for the ones protected by access policies.

Keywords: Privacy · Resource Sharing · Publishing · Authorisation · Access Control

1 Introduction

Usage control and data privacy is a main issue and we intend to simplify and ease the process of creating data on the web while giving users the chance to choose where newly created or published resources should be physically located, independently of where they are being used and by who.

While the term "publish" is intrinsically related "to disseminate to the public"[3], sharing is related to publishing privately. When a user uploads a resource to a domain on the Internet, the user is in fact publishing and establishing access policy rules upon that resource. Consequently, sharing a private resource is in fact the result of a more controlled resource "publishing" action.

Having in mind that one of the first principles of the World Wide Web (WWW) was to be a collection of hyperlinked resources, this is not always the case for private or access protected resources. This makes the latter second-class citizens of the WWW when it comes to being hyperlinked or embedded in different domains than the one where those resources are hosted on. Right now, most privately owned or shared resources are only accessible within the domain being hosted and by users enrolled on those domains that have been given access to the resource.

[3] http://www.merriam-webster.com/dictionary/publish

In order to refresh the concept of an Internet of hyperlinked resources, lets briefly examine the WWW evolution. While in the Web 1.0 era users mostly consumed data made available by companies, contribution of each user to the growing web of data was almost none or mostly limited to hosting their own personal web page. If another user wished to refer to any resource provided by others, s/he would create a link in their own homepage to refer to another resource that could be hosted on another domain, consequently creating a hyperlink in that web page and enriching the whole concept of a hyperlinked web.

At that time, proper authorisation controls were difficult to maintain and published data was mostly of public domain whereas servers did not provide appropriate authorisation access to resources and there were no fancy tools to organise uploaded resources like documents, photos or videos. Yet, users knew exactly where their resources resided and resources where mostly used in a hyperlinked fashion.

As the years went by and the WWW evolved to stage 2.0, users started to engage more on web applications and many stopped worrying about their own personally developed home pages. In this constant change and evolution, users were given a chance to start sharing resources among them and having better control over what and how they shared their resources.

The concept of their own home page was rapidly fading and switched for the one provided by one or more web applications of the most famous online companies (*i.e.* Facebook, Google +, tumblr., Microsoft Live, to name a few). These web applications are hosted inside a domain that is typically identified by a name[4] and each domain defines a realm of administrative autonomy, authority or control on the WWW. Such properties are barely transposed from one domain to another thus making each web application more distant and apart from each other.

The process of creating data and sharing resources became easier and much more simple than before but with a major trade-off as most of that sharing actions are only perceived inside a closed domain that acts as a data silo.

This is more or less innocuous when most of the users' social network remains on the same domain but quite the opposite when users register on multiple different domains across the network.

While users became more dependent on those big platforms, they also lost some of the control over their resources' hosting place and became limited to the rules imposed by each of those domains. That being said, most of the concept of an Internet made of inter-domain hyperlinks is becoming more and more disused as bigger companies thrive to keep all the resources within their domain premises (*e.g.* Facebook, Google, Apple). Of course hyperlinks are still used a lot, but most of them only relate to publicly published content (inside or outside the domain) and when data access policies need to be defined, it is only possible for the resources/users hosted/registered in the same domain.

For example, if examining Facebook's website, it is possible to perceive that it uses two different methods for users to show and share their photos. Users can

[4] http://en.wikipedia.org/wiki/Domain_name

opt for: (i) having their photos uploaded and hosted inside Facebook's website and sharing them on their profile page or (ii) simply by sharing external domain photos via a hyperlink on their profile page. When those photos are hosted on another web domain, Facebook automatically generates a preview/snapshot that is embedded on the profile page.

While both approaches have their downsides, in the former the user: may be lacking richer photo-editing features that can be found on more specialised photo-editing domains; is constrained to Facebook's access control policies for sharing that resource, thus restraining it from being shared to users not registered in the same web domain. On the latter approach, users can only perform sharing actions over photos that are publicly published on other web domains.

Each photo hosted on a different domain, which is managed by other access policies imposed by that domain, cannot be shared on the users' Facebook profile page. Thus being, a photo that is private or protected by any data access policy outside Facebook domain cannot be shared to others inside Facebook, because different domains apply different policy rules. While this approach is the one more alike the web of links that was sought with the WWW birth, different kinds of authentication and access policies prevent this kind of resource sharing from occurring. Also, if the hyperlinked resource is changed or becomes unavailable, the snapshot is not automatically updated; not reflecting the resource actual state and demonstrating that hyperlinking is not taking place in the real sense of the word.

By analysing how information resources are typically shared among users on the Internet, is was observed that privacy over shared resources typically only exists on closed domains, meaning that whenever a user shares a resource with any another user belonging to another domain, the security of that sharing is weak, most of the time only achieved if the owner sends the other user a very long Uniform Resource Identifier (URI), that is automatically generated by the domain where the resource resides. In terms of security and given enough time, any sequential code generator would be able to imitate the automatically generated URI for the sharing action thus allowing any user to access it. Despite this, anyone who ends up with that resource URI can actually gain access to it, therefore no authorisation access policies are defined over the resource.

The more data/resources a user uploads to the web, the more vulnerable the user becomes. If the user also needs to duplicate that data in several domains, not only the vulnerability risk increases as the coherence between copied data decreases if a mistake is made while copying that data/resources. Users should be able to upload a certain piece of data once and refer to it on other places in multiple occasions.

This work proposes an approach that consists in giving users the means for selecting in which domain the user's resources should remain (independently of where being used or linked) whilst managing their access control policies in a single place. It also enables a hyper-linkable web of inter-domain resources that can be restricted to Authentication, Authorisation and Accountability (AAA) as described on Fig. 1.

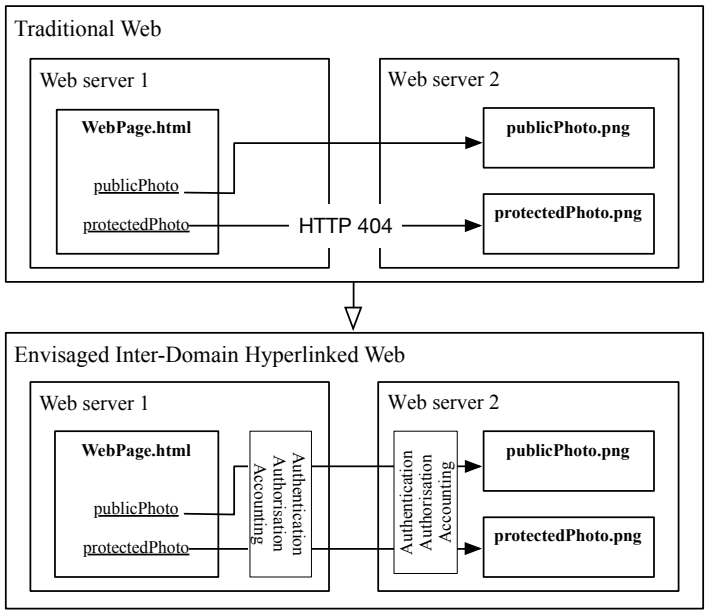

Figure 1. Traditional to envisaged hyperlinked web

After having demonstrated in previous work [4] how access control over each information resource [8] could be achieved in an inter-domain perspective, we intend to propose further enhancements to the existing framework to deal with some information resource sharing whilst keeping resources physically localised in user's preferred domains independent of where each resource is referenced.

For that, this proposal is achieved by reusing previous work [3] (described in section 3) and modifying one of its components, the Policy Enforcement Point (presented in section 4), nonetheless retaining important aspects of that system/framework like data provenance, trustworthiness of resources and making resources as private as possible (introduced in section 2). In the fifth section it is demonstrated how the prototype implementation was achieved and some unanswered/open questions are presented. Finally, the last section gives an overview of the proposed solution and suggests further research.

2 State Of The Art

In order to globally identify information resources, the URI [2] concept is widely used, allowing the establishment of relations between resources and users. The Resource Description Framework (RDF) [12] exploits this concept to form facts in triples (*i.e.* subject-predicate-object), where subject and predicate are URI, and object can be either a URI or a value (attribute).

Any information resource can be classified into three groups:

(i) **private**, when the resource is only accessible to the owner;

(ii) **protected**, when the resource can be accessed by the author and by another user. For a resource to be protected it needs to have been shared to another user by setting specific access control policies;

(iii) **public**, when the resource is publicly available to any user.

A private resource is automatically changed to protected when any other user besides the author has been granted access to it. Any private or protected resource can be taken to public state. Each state is not reversible, meaning that a protected resource cannot become a private resource and a public resource cannot become either private or protected.

Friend-Of-A-Friend (FOAF) [5] is a vocabulary devoted to describing people and their relations using a machine-readable format. Each FOAF file provides information about each user and can be used by websites to ensure users retain control over their profile information and relationships in a non-proprietary format. FOAF profiles can be used across different web applications to ensure users identify in a domain independently way. Nevertheless, FOAF does not provide a single user identity across different web domains. In order to reduce multiple user accounting across different web domains, FOAF+SSL [13] authentication protocol was created, providing a cross-domain protocol that intended to create a global decentralised authentication mechanism built upon the usage of FOAF profiles and easily adopted by web applications that support HTTPS protocol.

Lately, FOAF+SSL evolved to a W3C editor's draft[5] by the name of Web Identity and Discovery (WebID). This specification outlines a distributed and openly extensible universal identification mechanism, by combining asymmetric cryptography and Linked Data, that can make use of the FOAF vocabulary in the WebID universal identification mechanism, making it possible to link people and their profiles in either a public or protected way. In the work proposed by [14], by allowing delegation of access authorisation from a WebID to a third party it is shown how a web server or agent can act on behalf of its users.

Our previous work took this a step further by developing an Automatic Provenance Acquisition Framework [3] that also handles data provenance over user actions on the Internet, either privately owned or shared protected resources.

The PROV Ontology [9] uses the OWL2 web ontology language to express the PROV Data Model [11]. It can be used to represent and interchange provenance information generated in different systems and under different contexts.

[1] proposes a scalable and yet efficient storage model by exploiting structures of provenance logs and separating metadata from the generating process.

[7] not only addresses the creation of data but also proposes a capturing provenance model for information Web-based data access as well as information about the creation of data. Prizms Linked Data is a platform that creates datasets about the structural provenance of a host system to create provenance

[5] http://www.w3.org/2005/Incubator/webid/spec/identity/

leveraging [10] since it is still too difficult to publish and discover provenance in the Linking Open Data (LOD).

It is our belief that generating provenance data about a resource should be done as soon as possible, meaning that is should be recorded whenever a user performs an action over an information resource that could change its state and therefore we focus our work on generating data provenance for resources that are added or modified on the WWW, independently of being private, protected or public. Since provenance data is highly tied up with the resource, it can be private, protected or public according to the resource data access policies.

3 Background

In previous work [4], the authors proposed a decentralised architecture capable of providing authentication, authorisation and access control management based on provenance information. This framework proposes the use of action sensors (responsible for intersecting user actions on resources) and metadata generators for resource provenance information.

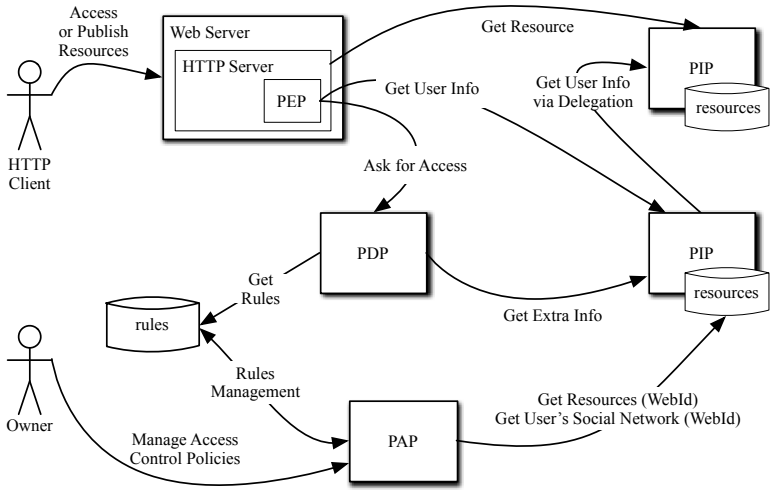

Figure 2. Framework architecture

The framework is mainly composed of four components. Fig. 2 depicts an overview of the entities and their interactions:

- **Policy Enforcement Point (PEP),** enforces authentication and guarantees authorisation controlled/authorised access to resources;
- **Policy Information Point (PIP),** is used to retrieve information related to users, resources and provenance information;

- **Policy Decision Point (PDP),** evaluates rules and policies that are used by the PEP;
- **Policy Administration Point (PAP),** enables users to build and manage access rules and policies over existing resources.

The PEP component provides means of validating FOAF+SSL authentication for every user, even if the user is not registered on the domain proprietary application. This component is responsible for tasks such as authentication, authorisation and gathering provenance information. It intercepts each request a server receives. When a request is intercepted, it uses the FOAF+SSL authentication in order to authenticate the user. If the requester is not able to provide credentials for FOAF+SSL, it bypasses the request directly to the web server application without interfering in the request process.

The PDP component is responsible for producing the decision of granting or not access to a resource, thus being responsible for checking if a given user should or not have access to it. The PDP receives user credentials from the PEP and according to the access policies, the PDP evaluates whether the requester should have access to the resource. If more information is needed to evaluate access, the PDP may contact any PIP to obtain related extra information.

Each PIP is responsible for obtaining information that is not available inside other components of the framework. This component acts as a broker interface to existing repositories that have information about resources, authors, web servers, and provides querying abilities to existing information repositories, therefore being able to answer queries like "return a list of all resources for a given user WebID" or "return a resource author". It also takes the role of a data provider and is capable of creating new provenance information (in the form of semantic annotations) about user actions and making it available on registered repositories.

Users configure access policies over resources and manage their relationships on the PAP. This component manages the user FOAF profile to maintain relationships up to date and provides the visual interface component where a user is allowed to manage access to resources. This component behaves like an access control management console, where it enables the user to control all his/her resources. At the time, access policies are defined as simple SWRL rules, but the framework allows any other kind of access policy models. This component uses existing PIPs to retrieve data about users, resources and relationships, so that access policies can be set upon them.

In order to address the issue of publishing privately, the team will use this framework as a starting point. The previously developed framework already enables authentication, authorisation and data provenance over resources in different web domains. All the work proposed required this existing framework for managing the users' authentication (via WebID), delegate policy decisions to domains where resources are hosted (taking policy rules to a fine-grained level) and to generate automatic data provenance information.

4 Proposal

In an Internet browsing scenario there are two possible ways for preventing or restricting users from accessing resources for which they have not been granted access. There is the client or server side approach.

As described in the previous framework, the Policy Enforcement Point component is responsible for acting as a broker between the user's request and the server response. Such component can be located either at the client side (by means of a browser extension) or at the server side (by means of a web application server module). Having adopted a server side approach in previous work, the same approach was preferred for the proposal of new features.

In order for this proposal to be effective, let's take into consideration that every information resource created or uploaded by a user is therefore physically hosted on a web domain but its URI can be referred in any other domain, independently of where the resource is hosted.

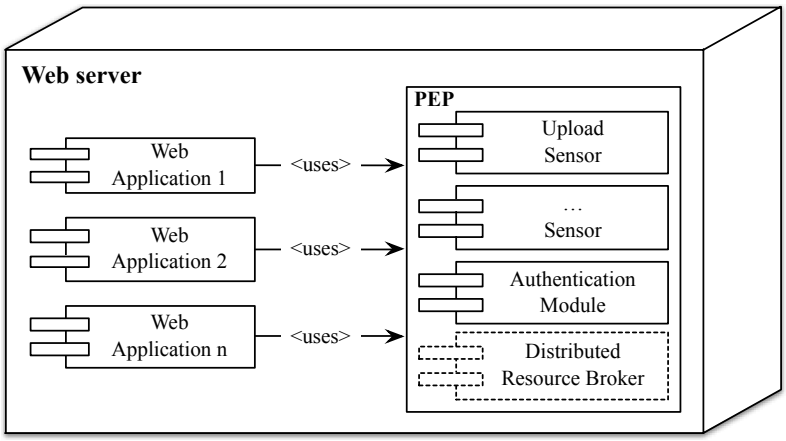

Figure 3. PEP: Distributed Resource Broker

By creating a Distributed Resource Broker inside the Policy Enforcement Point (on Fig. 3 represented as the dashed-line component), that is able to intercept a web resource request/response, it is possible to change and even override the information that is being sent on that request/response. While this is characteristically used by malicious groups in order to eavesdrop on requests, allowing network intrusion or man in the middle attacks, we intend to add new behaviour to this broker in order for it to provide a better and more decentralised control over the user's preferred hosting services.

Symptomatically, each existing web domain/application is responsible for hosting and keeping track of the resources it holds (*i.e.* independently of being

comments, documents, text, photos, movies, etc.) and the rules for displaying those resources are intrinsic to each of those domains not usually being possible to be reused or applied to users registered on other domains.

While the issue of handling the distributed authentication and authorisation over each of those resources has been dealt and solved in previous work [4] our intention is to enhance the decentralised resource management in order for resources to not have to reside on proprietary or isolated domain servers and still be capable of being accessed inside and outside those domains, by updating some of the components inside the web server without having to change all the web applications residing on top of the web server.

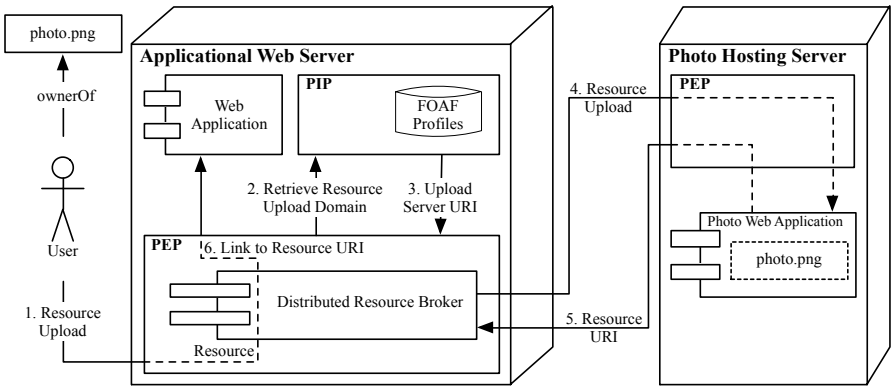

Figure 4. Resource upload

These new module's features primarily consist of firstly obtaining the appropriate service for hosting the resource, according to the resource type, thus allowing a resource to be hosted on different domains than the one where the resource is referred. Secondly, it filters the response by automatically loading any external resource, even from different web domains. This proposal addresses many of the afore mentioned issues in a way that would allow users to upload resources into one domain but instead of the resource being hosted in that domain it could be relocated to any other domain that could better support that kind of resource, thus increasing user's privacy over resources and the removal of duplicate resources and duplicate access policies. Such duplication of resources as well as access policies also increases the risk of jeopardising security, which is the opposite of what is being proposed.

Using this approach, any web page that makes use of resources that are available on domains using the framework, can be presented to final users with different views according to each user's access policies. That said, each individual user could eventually have a very different perception of the same webpage.

With these features, each individual user is given the chance to manage the list of preferred hosting services according to each distinctive type of resource. Such a list is based on the resource type being uploaded and based on that, the broker is responsible for accessing the user's preference list, retrieve the appropriate hosting service for that resource type, upload the resource to that hosting service, obtain the returned URI and use it in the actual domain doing the resource presentation as illustrated in Fig. 4.

When a user requests a web page that contains a link for the uploaded resource, the broker is responsible for providing the hosting service with any user credentials to prove that it is acting on behalf of the user's request, delegating any authorisation enforcement to the hosting service. If the user requesting the web page has any kind of access to that resource, the broker is responsible for retrieving it and rendering it on the webpage as depicted in Fig. 5.

When a user requests a web page with resources for which s/he has not been granted access, those resources (independently of where they are originally hosted) are automatically filtered out by the framework and will not be rendered on the web page.

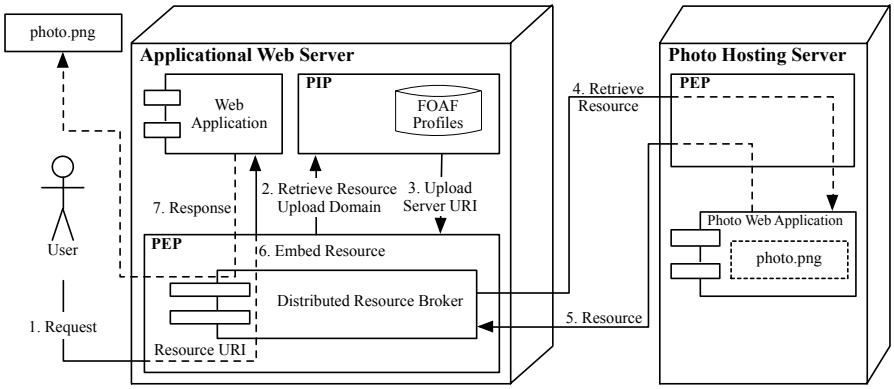

Figure 5. Resource request

As an example, lets depict a scenario where a user (U_a) uploads some resources (R_1 and R_2) to a web application and defines some access control policies over those resources, as depicted in Fig.6. When U_a requests the webpage via a browser, s/he is presented with a view similar to the one presented in the left in Fig. 7. If the resources' author (U_a) sets an access policy that grants view access to another user (U_b) to only one of those resources (R_1) the other user's (U_b) webpage rendering would be like the one in the centre of Fig. 7. Again, if the resources' author (U_a) gave viewing access to user (U_c) to the other resource (R_2), U_c's webpage rendering would be like the one in the right in Fig. 7.

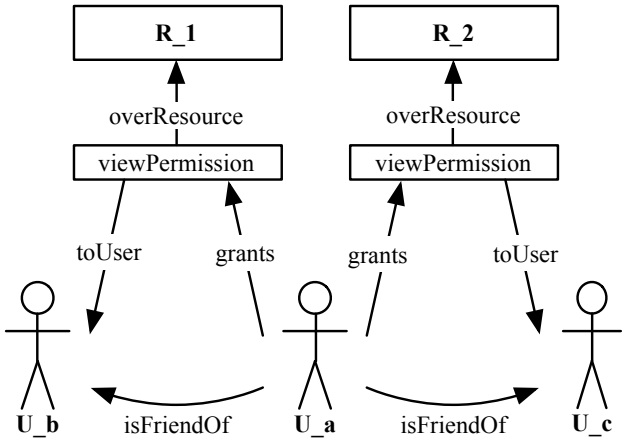

Figure 6. Resource access policy

Each web page is rendered solely and uniquely according to the credentials provided by the user requesting the webpage, which in the end, it can assume as many renderings as the number of permutations between the resources shown and the number of users whose been granted access to those. To increase security, access policy control is always enforced as close as possible to the resource being retrieved.

For a brief moment lets refer back to the envisaged hyperlinked web depicted on Fig. 1 and transpose that to the Facebook realm. Let's again consider a user (U_a) that sets some data access policies over some resources (R_1 and R_2) s/he owns on a certain domain appropriate to handling such type of resources (Flickr). Let's also consider the same user granted an access viewing policy to other users (U_b to R_1; U_c to R_2) as depicted in Fig. 6.

If U_a wished to display those resources on another domain (Facebook's profile page) but still enforce the original access policies and restrict unwanted access from others, we would only need to share a hyperlink to those resources (R_1 and R_2) on that domain (Facebook's profile page). If so, only users (U_b and U_c) would have viewing permission over each correspondent resource, and each one them would end up with different renderings of the same web page because data access permissions would still be enforced on the resource's original domain (Flickr).

As described, the same page may have multiple renderings depending on the users' data access permissions to each and every single resource. With minor effort and taking advantage of data provenance, it is possible for the resource's owner to have a log of its actions on the web as well as to provide means of trustworthiness to other users for all the published resources. This log also enables the user to check at any moment what data/resources s/he made available,

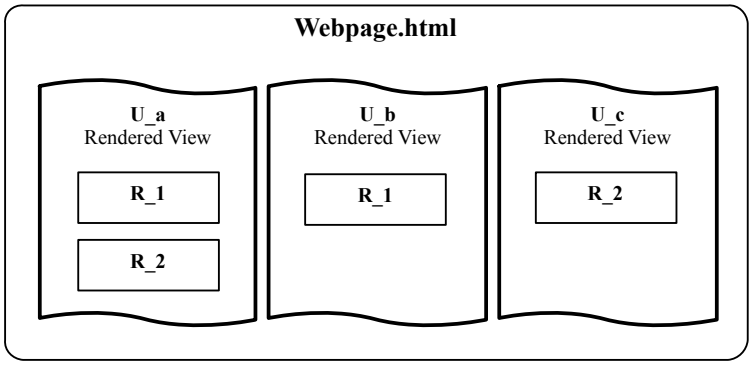

Figure 7. Web page rendering according to access policies

where these are hosted and who can access them, which is a proactive way of enhancing user's privacy on the WWW.

5 Discussion and Related Work

Testing a web server component on a big and fully operational website like Facebook, Google+ or any other of the kind was an unrealistic option. As a consequence, the previously proposed Distributed Resource Broker was developed as a module of the PEP component used in combination with the WordPress web framework [3] that had been previously engineering to accept WebID authentication. The introduction of the newly developed module was conducted on a web application server and several different WordPress domains were created to simulate several different hosting services for each of the different resource types.

For testing purposes, several different FOAF profiles were created to simulate different user identities. For each profile, several social relationships were added in order to recreate a social network. Each user maintained a blog in one of the several WordPress application servers where resources could be uploaded and shared. Each resource, according to its type (*i.e.* music, photo or video) was automatically uploaded by the framework to a specific WordPress instance only responsible for hosting those types of resources. After setting data access policies over the resources and relationships, users would only view other users blog's rendering according to the access policies that had been defined.

All the tests proved successful hence complying with all the goals proposed. Multiple web page renderings for the same web page were achieved due to the usage of different access policies for each user and uploaded resources were only physically located on the specific WordPress web server that had been set up for that purpose. Each WordPress blog website only hold blog entries and did not have any copy of the uploaded resources. While no major performance impact

was detected, one might suspect that in a true WWW experiment such impact might occur depending on the network performance provided by each of the individual hosting services being used.

Several open questions came up while studying and evaluating the issue of privately publishing resources on the Internet. We would like to share some of these questions, as it is our belief that they are fundamental for a better understanding of all the concerns that need to be overcome when using an approach such as the one proposed.

In this proposal we address only a range of available resources, those that are contained inside a file (*e.g.* photo, video, document, etc.). Sharing structured text (*i.e.* RDF, JSON, HTML) as a whole single indivisible resource is mainly the same as for documents and this approach is capable of handling those resources. To do that, only the resource URI is hyperlinked and the whole document is either embedded or downloaded on request. Nevertheless, privately sharing just part of a structured resource is a lot more difficult. While some resources are considered indivisible like (*i.e.* photos, proprietary formats), others can be divided into subparts thus allowing fine-grained access (*i.e.* HTML, RDF, etc.). This poses several difficulties namely:

- Do all structured resources have associated query languages or navigation utilities? While there are some query/navigation language like SPARQL for RDF resources, XPath for XML, JSONNiq for JSON, some structured documents (*e.g.* CSV, PDF) do not offer similar navigation thus making the document traversing more difficult, practically blocking the hyperlinking to just part of the document;
- How to specify the querying in the resource link and how to embed the query result in the web page referring that resource? How to handle semantic ambiguities resulting from hyperlinked resources?

Which component should perform the web page rendering, the web server or the client is also a relevant question. While either component is capable of handling that task, only development on the web server was conducted. Such component, when encountering a reference to an external resource that is protected by access policies, it delegates the request to the hosting domain which checks if the user making the request has access to the resource, thus rendering the result or completely removing it from the webpage before the request is sent to the requesting client.

When the result is not rendered, the requesting user is not even aware that s/he did not have access to a certain resource. Nevertheless, in a client-side implementation, if the client browser performed this external resource rendering, the user would have knowledge that a certain resource exists but that s/he does not have access to it. While the requesting user does not have access to the external resource, s/he actually knows about its existence, thus reducing the resource owner's privacy on the web.

Should it be possible to hyperlink a hyperlink (creating transitive links) or all hyperlinks should point to the original resource URI? Transitive links create

a daisy chain of references for the original resource. Therefore, when a request is daisy chained between multiple web domains, the users and each chain's web domain credentials must be delegated to the next element in the chain until the resource is encountered. Only then, the PEP is responsible for taking the decision of granting access or not. While delegating authentication and authorisation is not addressed in our proposal, we foresee the usage of the protocol proposed in [14], in order to handle authentication and authorisation in those daisy chained systems. This protocol proposes an extension to the WebID protocol, for allowing delegation of access authorisation thus allowing servers to act on behalf of users.

Priv.ly[6] is a product that emphasises on enforcing user's content privacy on the web in a way quite similar to our proposal. Its implementation is based on a client-side approach but only accessible for users that install a browser extension (available for Chrome, Firefox and Opera browsers). Our approach provides user's content privacy; inter-domain protected resource access control and sharing; and is a server-side implementation, more transparent to the final user, hence favouring a wider, quicker and rapid adoption by users.

Let's assume a typical use case registration in a web domain, where name, email and password are some of the commonly requested attributes to be filled in. Why should the user need to write his/her name when s/he should be given the chance to reuse the <foaf:name> property that resides in the identifying FOAF profile by simply hyperlinking to it? The domain where the registration is happening could eventually keep a cached record of the hyperlinked name, but should be able to display it correctly if the user ever changed it in his/her FOAF profile. This change would also automatically propagate to the dozens of other web domains where the user had performed the registration and the act of changing the name on the FOAF profile would be automatically reflected everywhere. The user name is only a possible example of such data re-usage and hyperlinking. Some approaches similar to this have been done for other types of documents, like XML resources. Enhancing the XLink specification [6] could make it suitable for usage inside the WWW.

6 Conclusions and Future Work

After conducting several tests on the system, with different types of resources, users, relationships and access policies, it was possible to identify that resources similar to closed and indivisible documents like music, photos or videos are best suited to be hosted outside a domain, because those resources can be easily rendered as a hyperlink. Images are an exception, as client browsers not only acknowledge the existence of a hyperlink, as well as they also display the image in the document itself. Hyperlinking to structured text or comments on forums or any other information introduced via a web form is more difficult to handle, as these are intrinsic to the domain where the text is introduced or the form is filled. Nevertheless, once the data/text is identified (by an URI) and localised (by an

[6] https://priv.ly

Uniform Resource Locator (URL), a form of URI), the resource's authorisation and controlled access can be managed, thus behaving like any other resource (*e.g.* photo).

Following previous research, users continue to be able to manage access policies over each resource in a single place. While hosting domains are responsible for enforcing any policy rules related to a single resource, presentation domains, where the resource URI is referred, are responsible for asking for access to those resources thus delegating the hosting domain with proper user credentials for it to enforce the authorisation process. This is a major advance because users are no longer forced to manage authorisation over their resources the way each domain wants and how it wants it. Furthermore, a user can share a resource with other users either on the same or other domains without having to duplicate the item and access policies. Before this proposal, what once was intended to be a single resource accessible by other users outside the domain, would end up being a duplicate resource on two or more different domains where access control would have needed to be duplicated and separately defined on each different domain.

Developed prototype and testing proved successful and it was possible to demonstrate that for a one time uploaded resource, it was possible to reference it on different web domains while still preserving access control policies over it. One overall advantage is that users can even keep a log of their every data creation and publishing on the web.

Trustworthiness also plays a very important role when creating a hyperlinked web like the one proposed. Just because a user has access to a resource, does not mean that s/he wished to have it rendered. If a malicious script or document exists on a referred external resource, and the user credentials allow access to that resource, this could pose several security threats. What has been proposed could be enhanced with a user/resource blacklist, in order to filter which resources could be loaded based on the referring resource or even resource author. If the user does not trust another user or a specific resource, either the user or specific resource could be blacklisted.

Unfortunately, most of these advantages come with a price and may not be achievable at the moment. There are certain technological problems that still constitute an obstacle to a full adoption of such a hyperlinked web of resources. The number of requests between web servers would certainly increase and so the available bandwidth would decrease, reducing Internet browsing speeds.

Setting aside technological difficulties, we believe that in the future, every information resource can be hosted in a different web domain than the one where the resource is being referenced. Yet, for HTML web page rendering, new tags or enhancements to existing ones will need to be introduced in current vocabularies for referring to external resources and that itself is a new research field.

In conclusion, it was proven that (i) by extending an existing framework it is possible to publish resources privately by only developing a Distributed Resource Broker service that resides on the application web server; (ii) privacy over user resources is enhanced, because resources can be hosted exactly where the user feels most comfortable with and relies on; (iii) trustworthiness of a resource is

given by the user's association to the resource, backed up by provenance data provided by the framework.

Acknowledgements. This work is supported by "Fundo Europeu de Desenvolvimento Regional (FEDER)" funds through the "COMPETE - Programa Operacional Factores de Competitividade (POFC)" program, under the project Ambient Assisted Living for All (AAL4ALL - QREN 13852).

References

1. Jemal H. Abawajy, Syed I. Jami, Zubair A. Shaikh, and Syed A. Hammad. A framework for scalable distributed provenance storage system. *Computer Standards and Interfaces*, 35:179–186, 2013.
2. Tim Berners-Lee, Roy T Fielding, and Larry Masinter. Uniform Resource Identifier (URI): Generic Syntax. http://www.ietf.org/rfc/rfc3986.txt, 2005.
3. Nuno Bettencourt, Rafael Peixoto, and Nuno Silva. Automatic Traceability Acquisition Framework. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics (WIMS'2012)*, New York, USA, June 2012. ACM Press.
4. Nuno Bettencourt and Nuno Silva. Recommending Access to Web Resources based on User's Profile and Traceability. In *2010 10th IEEE International Conference on Computer and Information Technology*, number Cit, pages 1108–1113. IEEE, 2010.
5. Dan Brickley and Libby Miller. FOAF Vocabulary Specification, 2010.
6. Steven J DeRose, Eve Maler, David Orchad, and Norman Walsh. XML Linking Language (XLink). http://www.w3.org/TR/xlink11/, 2010.
7. Olaf Hartig. Provenance Information in the Web of Data. *Proceedings of the Linked Data on the Web LDOW Workshop at WWW*, 39(27):1–9, 2009.
8. Ian Jacobs and Norman Walsh. Architecture of the World Wide Web, Volume One, 2004.
9. Timothy Lebo, Satya Sahoo, and D McGuinness. PROV-O: The PROV Ontology. http://www.w3.org/TR/2013/REC-prov-o-20130430/, 2013.
10. Timothy Lebo, Patrick West, and Deborah L. McGuinness. Walking into the Future with PROV Pingback: An Application to OPeNDAP using Prizms. In *Proc. of the 5th International Provenance Annotation Workshop (IPAW'2014)*, 2014.
11. Luc Moreau and Paolo Missier. PROV-DM: The PROV Data Model. http://www.w3.org/TR/prov-dm/, 2013.
12. Mark H. Needleman. RDF: The resource description framework. *Serials Review*, 27(1):58–61, 2001.
13. Henry Story, Bruno Harbulot, Ian Jacobi, and Mike Jones. FOAF+SSL: RESTful Authentication for the Social Web. *Current*, pages 1–12, 2009.
14. Sebastian Tramp, Henry Story, Andrei Sambra, Philipp Frischmuth, Michael Martin, and Sören Auer. Extending the WebID Protocol with Access Delegation. In *Third International Workshop on Consuming Linked Data (COLD2012)*, 2012.