2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing

Open Social based Collaborative Science Gateways

Wenjun Wu Hui Zhang ZhenAn Li
State Key Software Development Environment Lab
Beihang University
Beijing,China
{wwj,hzhang,lizhenan} @nlsde.buaa.edu.cn

***Abstract*—In data-driven science projects, researchers distributed in different institutions often wish to easily team up for data and computing resource sharing to address challenging scientific problems. Typical VO based authorization schemes is not suitable for such a user organized scientific collaboration. Using the emerging OAuth protocol, we introduce a novel group authorization scheme to support ad-hoc team formation and user controlled resource sharing. Integrating this group authorization scheme, we define an OpenSocial based scientific collaboration framework and develop a science gateway prototype named as Open Life Science Gateway (OLSGW) to verify and refine the framework. Our experience with development of the OLSGW shows that OAuth 2.0 based group authorization scheme is a very promising approach to resource sharing in Cloud environments, and the OpenSocial based framework can facilitate science gateway developers to create domain-specific collaborative applications in a very flexible way.**

Keywords-Science Gateway; Collaboration; Authorization; OpenSocial

I. INTRODUCTION

A Science Gateway [1] is a computational web portal that includes a community-developed set of tools, applications, and data customized to meet the needs of a targeted community. By enabling data-driven scientific computing such as scientific simulation, data analysis and visualization, it becomes a collaborative cyber-environment on which researchers working on the same or close domains can easily team up to address challenging scientific problems and elevate experimental datasets to innovative theories. Especially, the OpenSocial [2] networking framework presents a powerful collaboration model for facilitating scientists to manage their virtual research enterprises and share computational resources ranging from large-scale datasets to high performance clusters.

The key notion related to scientific collaboration is virtual community or virtual organization (VO) which is a collection of individuals and institutions established according to a set of resource sharing rules. A virtual organization allows people to access virtually shared resources physically distributed in different places. The core function in VO management is a community authorization service that maintains VO membership and enforces fine-grained access control policies specified by community users. Most existing VO authorization frameworks such as CAS [3] and VOMS [4] based on PKI infrastructure are usually designed to support inter-institutional trust relationship formed by a centralized authorization server and local admission point from institutional resource providers. Any user who wants to access shared VO resources must have a valid identity trusted by all the service points and obtain a permission decision from the authorization server. There is no easy way for individual users instead of VO administrators to describe relatively casual and temporary trust agreements among research groups whose member don't have a valid and long-term identity for the participating resource providers.

In contrast, social networking and related security technologies greatly simplify web resource sharing, which makes it possible to find a more light-weight solution to the formation of impromptu scientific collaboration. Based on the emerging open authentication and authorization protocols such as OpenID [5] and OAuth [6], we introduce a novel group authorization scheme to support ad-hoc team formation and resource sharing for scientific collaboration across institutional boundaries. This scheme doesn't rely on institutional identity and authorization infrastructure but focus on social grouping and user controlled access. Through the scheme, users can create, manage their research teams and control resource sharing without involvement of system administrators.

Moreover, we integrate the user-driven group authorization scheme into the OpenSocial framework to build up a scientific collaboration platform on which science gateway developers can create domain-specific collaborative applications. OpenSocial was initiated by Google as a web 2.0 approach to integration of web applications and building collaborative cyber-environments. Within the framework OpenSocial, every web application is regarded as a gadget that defines its HTML content and control logic in client-side JavaScript. Through the social data API of OpenSocial, a gadget can retrieve social graph information about people, their friends, and their groups. By combining these OpenSocial APIs with computational web-services APIs to Cloud and Grid resources, we leverage the OpenSocial framework to support data sharing, collaborative data analysis and knowledge discoveries. Using the APIs from this OpenSocial based scientific collaboration framework, Science Gateway developers can easily improve user experience, streamline scientific application management, and promote scientific productivity for their science domains.

978-0-7695-4395-6/11 $26.00 © 2011 IEEE
DOI 10.1109/CCGrid.2011.43

This paper is organized as follows. Section 2 introduces the related work and discusses the key design issues in supporting ad-hoc resource sharing. Section 3 describes the OpenSocial based scientific collaboration framework and demonstrates a prototype named Open Life Science Gateway (OLSGW) as an example for the framework in detail. Section 4 compares our work with related research projects. The conclusion and future work are given in Section 5.

II. GROUP MANAGEMENT AND ACCESS CONTROL

Nowadays, it is common to have multiple researchers from distributed institutes to work together in the same scientific research projects. They have to share their experiment data, run simulations jointly, and analyze results collaboratively through a computational cyber environment that integrates distributed and heterogeneous computing resources. To achieve such a virtual computational organization for a collaborative scientific research, two important tasks are needed: group management and access control. Group management facilitates scientists to quickly develop a collectively working relationship in the environment that authorizes them to access the aggregation of resources following a fine-grained access control policy. This section compares both VO based group management and social networking based solution.

A. VO Based Group Management

Over the past decade, numerous research efforts have been done on virtual organization in terms of group management and access control. Well-known examples include Globus Toolkit Grid Security Infrastructure (GSI) [7] which was initially designed for authentication and delegation. Based on the GSI, two major VO management schemes such as CAS and VOMS were proposed and implemented to support VO group management, user access control policies and agreement between VOs and resource providers. In Both schemes, a centralized VO server uses either LDAP or a relational database to keep track of the VO group membership and authorization rules based on user capabilities and roles in the VO. A user wishing to access VO resources has to contact the VO server, which makes authorization decisions whether to delegate access privileges to the user based on his role and associated control policies. Administration tools allow VO administrators to create groups, set user attributes and define authorization rules. The difference between CAS and VOMS is about how to organize the VO authorization data and implement user attributes in user credentials.

The major problem of the frameworks comes from the basic design idea that heavily focuses on virtual organization across the boundaries of security domains. The PKI trust framework based on which these authorization mechanisms were developed, has to be maintained by IT administrative staffs from institutions such as universities, research labs and government agencies. Every user must have either some sort of federated identification or local ID to request access permission from his community Policy Decision Point (PDP) and show the permission to Policy Enforcement Points (PEP) of resource providers. Obviously these schemes seem too height-weighted when they are applied in ad-hoc collaboration scenarios.

Imagine the following use case: User Alice who has a large amount of simulation data in her local storage needs to share with her collaborator Bob because Bob have access to a high-end computing resource needed for data analysis. Since their resources are managed by two different institutions, apparently they need a virtual group across the institutions to share the computing, storage and data resources. To support such an across-domain research group, these institutions have to set up a PKI enabled VO, and have Alice and Bob's group created in the VO authorization database beforehand. Moreover the VO management usually only allows VO administrators to create groups, add users to groups and set users' attributes. Regular users like Alice and Bob have no privileges to initiate their virtual group, define authorization policies, or invite other researchers from the outside into the group dynamically.

B. Scientific Team Management Based on Social Network

We propose a new scientific team management scheme using social networking to support dynamical and ad-hoc science collaboration. A scientific team often comprises a science social group and shared resources including computing utilities and datasets. A science social group is a collection of researchers who are willing to collaborate on a non-trivial science project. These researchers may contribute a variety of dynamic resources into this group to establish a virtual resource pool that is available to all of them. Our scientific team management scheme provides science social group management for the researchers, and supports them to control access to the group resources in the virtual pool. Figure 1 displays the basic modules in the team management scheme. The group authorization manager stores the group membership, user-defined access control policy and the virtual resource pool.

*1)Science Social Group Mangement:*The group authorization manager allows users to create their own interest groups and invite their friends to join. Through the public social graph APIs, the group authorization manager can import the friend list of each user under their permissions from public social network sites such as Google, and construct a group candidate list following friend-of-friend closure and interest profiles. The science group creator who becomes the group administrator by default can choose people from the candidate list and send them invitations to join in the group. Each group member may have many attributes describing their capabilities and roles in this group. The group administrator can set these attributes dynamically based on the progress of the research projects and the requests from his team member. The group

authorization manager stores all the membership data including group information, team members and their attributes in a relational database that also acts the local cache for imported social graph data for users.

*2)User controlled access to group resources:*Each group has a resource list that is a collection of access URLs to web-enabled resources such as online storage space, active cloud clusters as well as scientific web-services. In this paper, we assume that each resource should have either a RESTful access interface or Web-Services SOAP interface, which can be represented by URL links. Every user can voluntarily register his resources in the resource list and describes the resource interfaces. Every resource contributor can also specify the access control policy for his resource. For example he may want some team members with senior roles to have full read and write privilege on his own data set, while other members only have read permission. The group authorization manager follows the user defined policy and makes decisions whether a team member request should be granted or rejected. In this way, the authorization manager enables the resource owner and resource requestors to create a delegation relationship illustrated in the Figure 1.

From Step 1 through Step 3, when a user registers his resources with the resource list, he delegates the authorization right to the group authorization manager (PDP) by configuring group access policies and presenting an access token which can be accepted by the resource provider acting as PEP. At Step 4-5, when a team member makes a request for resource access, the group authorization manager checks the owner-specified policies and decides whether to grant the access token to the requesting member. At Step 6, the member finally obtains a valid OAuth access token to visit the shared resource. Note the access token can be a long-term credential or a short-time one according to the owner's requirement. For a short-time one, the group authorization manager can either refresh the access token on behalf of the resource owner or ask the owner to renew the token if the token is expired. This renew procedure is illustrated at Step 7.

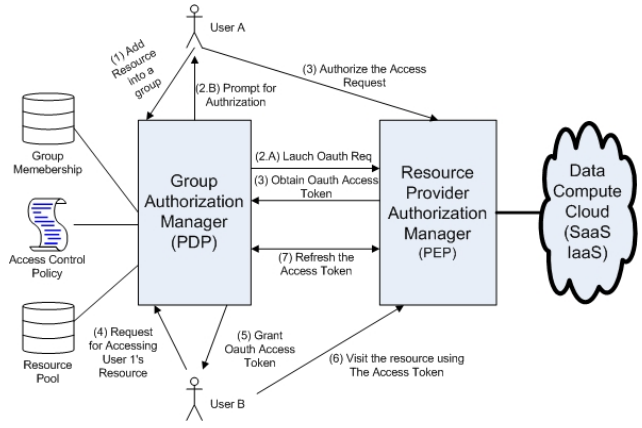

Figure 1. User controlled access to group resources.

III. OPEN SOCIAL BASED SCIENTIFIC COLLABORATION FRAMEWORK

Google's OpenSocial framework standardizes the practices of both gadget and social-networking sites, enabling web developers to write gadgets with social capability that can run in any OpenSocial compliant environment. More importantly, OpenSocial supports the OAuth protocol to enable externalization of application services to web gadgets without forcing users to expose their passwords and other credentials to gadget hosting environments known as gadget containers. Based on the OpenSocial group interface and OAuth support, we can implement our scientific team management scheme inside an OpenSocial container. Further, by combining the OpenSocial APIs with OAuth-enabled web-services APIs to Cloud and Grid resources, we leverage the OpenSocial into a scientific collaboration framework shown in Figure 2. The implementation of our framework is done on the basis of an OpenSocial reference implementation named Shindig [8], which is an Apache project with a rapidly emerging developer community. Note Shindig is not a full-fledged OpenSocial container because at the moment of writing, Shindig still has no services such as gadgets layout, gadgets management and security. OGCE gadget container implements an iGoogle like layout and adds OpenID support to enhance the Shindig container [9]. We decided to implement our framework based on their container code.

A. Open Social Collaboration Framework

As shown in Figure 2, there are four major server-side components in the framework, including the social data server, group authorization manager, service connector and gadget server. The social data server collects social graph data for each user and provides application gadgets with the interface to retrieve and update these data. This social data API is defined on the model < person, group, activity, appdata >, which can describe the profile information for a person, relationship between people as well as activities. Web gadgets can retrieve the social data through the API to implement social applications. The group authorization manager maintains the group membership and user-specified access control policies, and acts as a PDP for granting access to shared resources. The OAuth-enabled service connector is a mediator among the group authorization manager, web application gadgets and remote service providers. For contacting each remote OAuth-enabled resource provider, the service connector launches a service proxy as an OAuth consumer to complete OAuth protocol flows and invoke remote services.

In the OpenSocial, every web application gadget encapsulates its HTML content and JavaScript control logic in a single XML file. The gadget server parses this XML file and dynamically generates a gadget object that can be rendered in the browser-side OpenSocial environment. Because of the security sandbox in a Web browser, a gadget can't make cross-domain JavaScript calls to communicate

directly with remote service providers. Only through a service proxy created by the service connector, the gadget can access the remote resource via OpenSocial gadget IO library. The proxy can handle both RESTFul content requests and RPC calls over JSON.

B. OAuth based Group Authorization Mechanism

The group authorization scheme discussed in Section 2 depends entirely upon IETF Open Authorization (OAuth) standard, which was proposed to address the authorization delegation problem for web mashup sites that always need to access users' web resource located in other web servers. A three-legged OAuth protocol involves three parties: the User, OAuth Consumer and OAuth Provider.

OAuth Authentication is done in three steps:

1. The Consumer obtains an unauthorized Request Token from the Provider.
2. The User authorizes the Request Token through the Provider.
3. The Consumer exchanges the Request Token with the Provider for an Access Token.

By authorizing the request token for the provider to grant the access token, the user can easily control access to his owned resources, which is very essential for our science team management scheme. The original OAuth 1.0 protocol was primarily designed for web browsers and didn't provide profiles to support desktop and mobile devices. OAuth 2.0 [10] is the next evolution of the OAuth protocol, which greatly extend the client profiles by providing specific authorization flows for web browsers, desktop applications, and smart phones. More importantly, OAuth 2.0 introduced a long-lived refresh token that can be used to renew an access token with limited lifetime. In OAuth 1.0, the lifetime of an access token is only set by the provider. The consumer has no way to renew the token after it is expired. So in a real application environment, the provider would like to issue a long lasting access token, typically valid for a year or unlimited lifetime to avoid repeated initiation of OAuth authorization flows for improving user experience. But such a practice could lead to potential security issues. OAuth 2.0 introduces the renewal procedure that allows an OAuth consumer to hold a valid access token for a long time without bothering the user for the permission, and keep the access token limited under the control of the user.

In our framework, the service connector creates a service proxy as an OAuth consumer, which communicates with remote OAuth providers. Whenever a user registers a new resource in the resource list, the group authorization manager will contact the service connector to launch a new service proxy for connecting to the remote resource provider according to OAuth 2.0 protocol flows. Under the control of the group authorization manager, the service proxy sends token requests to the provider, obtains an access token from the provider, and renews the access token when necessary.

Although OAuth2.0 was only proposed last year, there are already some open source OAuth2.0 libraries and some OAuth2.0 compatible cloud sites including FaceBook, Twitter and SalesForce [10]. Some features in our framework such as the service proxy have been available in the Shindig package for a while. Moreover, the latest Shindig package already added support for OAuth 2.0 including the new header parser and protocol handler, which enables us to fulfill our scientific team management scheme on the basis of the Shindig system. First, we extended the database schema in the Shindig package to incorporate the group membership and access control policy. Second, we extended the service proxy in the Shindig to support our service connector. Finally, we developed a new software module for the group authorization manager, which can interact with the enhanced proxy code and provide a RESTful interface to the group management service.

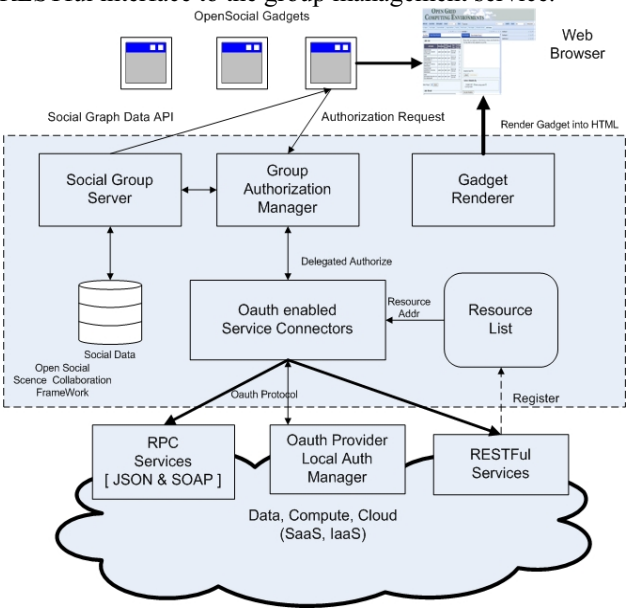

Figure 2. OpenSocial based Scientific Collaboration Framework.

C. Collaborative Open Life Science Gateway

At present, we are testing and refining the Science Collaboration Framework in a real system -- Open Life Science Gateway (OLSGW) [11]. It is an integrated cyber computational environment for the Life Science community to utilize the Cloud and Grid resources for computing and data management. OLSGW hides the complexity of using these resources from users by running reliable workflow management middleware and providing the community with web 2.0 gadgets and an easy-to-use web interface. Currently OLSGW has integrated a group of protein sequence analysis tools ranging from sequence search to protein structure prediction in order to deliver them to life scientists in Software-As-A-Service (SaaS) paradigm. We split the original version of the OLSGW package into the SaaS part and the UI part. The SaaS part was a standalone package including the bioinformatics application software packages and corresponding web-services wrappers. Users can install an instance of the OLSGW SaaS package if their

resource providers don't provide OAuth-enabled web-services interface. For the OLSGW UI part, we refactored the OLSGW gadgets to make them become social-aware so that they could display group activities and analysis results. Figure 3 shows a collection of collaborative OLSGW gadgets in the container for the collaboration framework.

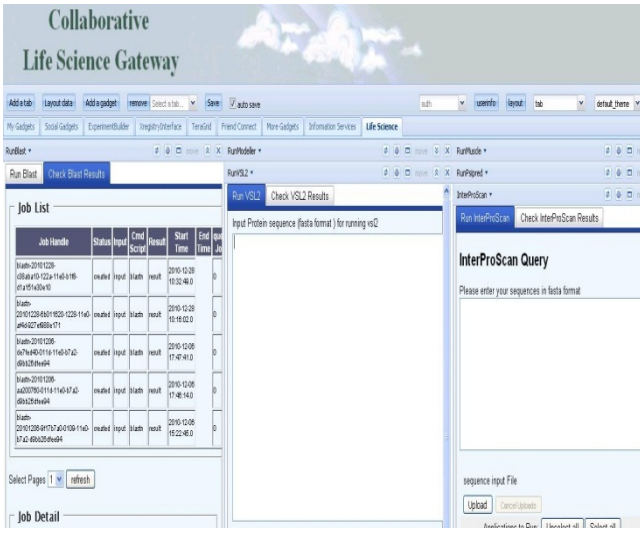

Figure 3. Collaborative Open Life Science Gateways.

During the code refactoring of the OLSGW, we found out that although most features in the OpenSocial data API have been developed in the Shindig, the group API was not well implemented. To gadget developers, it is a group of RESTful and JSON-RPC services that expose the social data graph in the database of the server. Ideally we just need to add the database connector code into the Shindig package to query the user and group information in the group membership database. However, Shindig doesn't implement the Group according to the OpenSocial specification in the general sense but only provides the special group named "Friend" for gadgets. Apparently the "Friend" group is too general to describe specific science teams in the OLSGW. Therefore we extend the implementation of the social data part in Shindig, and add more features such as Workflow based on the requirement of the OLSGW groups and activities to the JavaScript APIs.

Table 1 lists the RESTful API of the OLSGW social data features:

TABLE I. OLSGW SOCIAL DATA API

Type	Restful URL	Explanation
Group	/people /{userid} /@all-groups	List all the group for this User
	/people /{userid} /{groupid}	List all the people connected to the user {userid} in group {groupid}
Activities	/activities /{guid} /@self	Collection of activities generated by given user
	/activities/{guid} /@self/{appid}	Collection of activities generated by an app for a given user
	/activities /{guid} /{groupid}	Collection of activities generated by an app for a given user in a given group
	/activities/{guid} /{groupid} /{appid}	Collection of activities generated by an app for people in group {groupid} belonging to given user {uid}
Work flows	/workflow /{userid}	List all the workflows created by the user {userid}
	/workflow /{userid} /{appid}	List all the workflows of the application {appid} created by the user {userid}
	/workflow /{userid} /{groupid}	List all the workflows created by the users for the experiments in the group {groupid} in which the user {userid} joins
	/workflow /{userid} /{groupid} /{appid}	List all the workflows of the application {appid} created by the users for the experiments in the group {groupid} in which the user {userid} joins

Collaborative OLSGW application gadgets can be developed based on these social data API. For instance, if members of a research team want to share their active workflows in the OLSGW, they would like to be aware of any analytical workflows performed on the sharing dataset. This requires OLSGW gadgets to present a workflow history that reports not only current user's workflow records but also the workflows run by his teammates.

A collaborative BLAST gadget can be a good example of how all the components in OLSGW fit together. As the use case discussed in Section 2.1, OLSGW user Alice and Bob wish to share both computational resources and experimental datasets. Suppose Bob has installed a BLAST program on his local cloud environment that has an OAuth protected RPC services for launching computational jobs and Alice has put a large amount protein sequence files on a Web storage space that also supports OAuth. Through OLSGW group gadget, they can create a science group and add their OAuth contact addresses. Running at the backend of the OLSGW Shindig server, the group authorization manager and service connector establish communication channels between application gadgets and registered cloud resources. The BLAST application gadget can pull the group information through OLSGW group RESTful API, download the sequence data from Alice's web storage server, and launch BLAST jobs on Bob's local cloud cluster. The gadget can also record the activities of every team member and display their workflows to their collaborators

so that everyone knows what others are doing with the dataset.

IV. Related Work

Research efforts have been made to investigate the synergy between cloud computing and social networking. For example, social VPN [13] is introduced to enable users to create peer-to-peer overlay network though a friendly interface running in the infrastructures of social networking sites such as FaceBook and LinkedIn. Social cloud is proposed in [14] to dynamically provision storage resources contributed by users through friendship. Both works developed their own FaceBook applications that act as a social venue to support authentication and group management in the infrastructure of FaceBook. None of them mentioned about OAuth2.0 based authorization delegation mechanism between their FaceBook applications and resources contributed by users. In contrast, our research intends to create an in-house social networking platform that can interact with other commercial social networking sites and presents an OAuth2.0 group authorization scheme to support user controlled resource sharing.

Popular science social network sites such as MyExperiment [15] and nanoHub [16] have been successfully serving a variety of science communities for sharing computational workflows and running data analysis on their Cloud and Grid resources. But to my best knowledge, they built proprietary social infrastructures without using open social networking protocols like OAuth and OpenSocial. Interestingly, from the online document of MyExperiment, an OAuth1.0 based API can be found, which allows developers to create their applications to access MyExperiment APIs. Such a consumer-side OAuth API can be very beneficial for our OLSGW users. If MyExperiment could upgrade their OAuth support to the new version OAuth 2.0, MyExperiment service would become a resource provider for our OLSGW platform. If an OLSGW user has bioinformatics workflows stored in the space of MyExperiment, he can access their workflows without visiting the MyExperiment website and even share these workflows with his teammates through OLSGW grouping mechanism.

V. Conclusion

In this paper, we introduce a novel group authorization scheme to support ad-hoc team formation and user controlled resource sharing for scientific collaboration across institutional boundaries. We also describe the OpenSocial based scientific collaboration framework that integrates the user-driven group authorization scheme to build up a scientific collaboration platform on which science gateway developers can create domain-specific collaborative applications. Based on the framework, we have been refactoring an Open Life Science Gateway and transforming it into a collaborative computational cyber-environment for the Life Science community for computing and data management. So far we have completed the collaborative BLAST gadget to show group workflows. We are planning to add more support for data sharing by integrating the data model into the RESTful social data API. Our experience with development of the OLSGW shows that the OAuth 2.0 based group authorization scheme is a very promising approach to resource sharing in Cloud environments, and the OpenSocial based collaboration framework will facilitate developers to create their Web 2.0 science gateways with more flexibility.

Acknowledgment

This work is partially supported by a grant from the Major State Basic Research Development Program of China (973 Program) (No.2005CB321903) and grants from the National High Technology Research and Development Programs of China (863 Program) (No. 2009AA043303).

References

[1] N. Wilkins-Diehr, D. Gannon, G. Klimeck, S. Oster, S. Pamidighantam, TeraGrid Science Gateways and Their Impact on Science, IEEE computer Nov 2008.

[2] OpenSocial Specification, http://www.opensocial.org/.

[3] L. Pearlman, V. Welch, I. Foster, K. Kesselman, S. Tuecke, A community authorization service for group collaboration, IEEE Workshop on Policies for Distributed Systems and Networks, 2002.

[4] R Alfieri, R Cecchini, V Ciaschini, VOMS From gridmap-file to VOMS: managing authorization in a Grid environment, Future Generation Computer Systems, 2005.

[5] OpenID, http://openid.net/.

[6] OAuth (RFC 5849), http://tools.ietf.org/html/rfc5849.

[7] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, A security architecture for computational grids, Proceedings of the Fifth ACM Conference on Computer and Communications Security Conference, 1998, pp. 83–92.

[8] Shindig, http://shindig.apache.org.

[9] Z. Guo, R. Singh, M. Pierce, Building the PolarGrid Portal Using Web 2.0 and OpenSocial, In GCE '09: Proceedings of the 5th Grid Computing Environments Workshop, pages 1–8, 2009.

[10] OAuth 2.0, http://tools.ietf.org/html/draft-ietf-oauth-v2.

[11] W. Wu, R. Edwards, I.R. Judson, M.E. Papka, M.Thomas, R.Stevens, TeraGrid Life Science Gateway, Proceedings of the 2008 TeraGrid Conference.

[12] SalesForce, http://cloudcomputing.sys-con.com/node/1640061.

[13] R. Figueiredo, P. O. Boykin, P.S. Juste, D. Wolinsky, Social VPNs: Integrating Overlay and Social Networks for Seamless P2P Networking, IEEE WETICE/COPS 2008.

[14] K. Chard, S. Caton, O. Rana, K.Bubendorfer, Social Cloud: Cloud Computing in Social Networks, 2010 3rd IEEE International Conference on Cloud Computing.

[15] MyExperiment, www.myExperiment.org.

[16] nanoHub, www.nanoHub.org.