2013 IEEE 9th International Conference on e-Science

Malleable Access Rights to Establish and Enable Scientific Collaboration

Ferry Hendrikx and Kris Bubendorfer
School of Engineering and
Computer Science
Victoria University of Wellington
New Zealand
Email: ferry,kris@ecs.vuw.ac.nz

***Abstract*—**

Collaborative systems require access control to prevent unauthorised access and change. Access control has a number of issues, including administration and maintenance overheads. In this paper we argue that it is time to reconsider how access controls work, particularly with scientific and data related domains, and to this end we propose a new paradigm based on a user's demographics and behaviour, rather than simply their identity. In essence, it is both who you are and what you do that is important. We introduce GRAft, our Generalised Recommendation Architecture that allows us to support a range of different recommendation models, and provide case studies to illustrate the usefulness of our architecture.

I. Introduction

The Internet has seen significant growth over the past decade [1], [2], and even the more modest growth rates seen recently [3] point to an increasing number of participants and usage. Since the advent of Web 2.0, the trend is towards using collaborative systems to meet, organise, discuss and cooperate. Within an organisation this improves communications and decision making, and reduces both time and cost [4] as it allows an organisation to apply the best people to any given situation. Collaborative computing (hence collaborative systems) encompass "the use of computers to support coordination and cooperation of two or more people who attempt to perform a task or solve a problem together" [4].

As with any computer system, collaborative systems require access control (and auditing) to prevent unauthorised access and change. Access control is defined in [5] as a process that requires "every access to a system and its resources be controlled and that all and only authorised accesses can take place". Access control policies can be classified into three groups: Discretionary, Mandatory and Role-Based [5], [6].

- **Discretionary Access Control** (DAC) policies restrict access to an object based on the identity of the current user and a set of rules [6]. These systems typically assume that all objects have an "owner" [7], and users may reassign their privileges to others [5]. Privileges to an object are assigned to users when they create an object, and can be changed or revoked by an administrator.
- **Mandatory Access Control** (MAC) policies restrict access to objects using a global policy. The policy is centrally controlled by an administrator, and cannot be changed by the users. Every user and object is given a security classification. Users must have the same or higher security in order to read an object with a given security classification [6].
- **Role-Based Access Control** (RBAC) policies are based on the concept that access to an object is limited to certain roles that have been defined within an organisation. The responsibilities and qualifications of any particular user determine the roles that are assigned to them [5]–[7] and hence which objects they may access.

A number of collaborative services are defined in [4]. This paper will examine and utilise three of those services: Forums (called Bulletin boards in the original paper), File and document sharing, and Document management.

- **Forums** allow for online discussions and interactions between participants with a similar interest. The main drive for a forum is the exchange of ideas, whether it be for problem solving, discussing politics or collaborative work between academics. All posts to a forum are effectively read-only, with each participant adding new content to the discussion. Off-topic posts can be ignored, while offensive ones can simply be removed by a moderator. Forum access controls are often based on a simple RBAC scheme: users acquire a *participant* role, which allows them to take part in the forum. A separate *administator* role is assigned to the forum owner, who maintains the forum. This user may promote other users by assigning them a *moderator* role.
- At its most basic, **File and document sharing** allows for the simple sharing of files between users. The users work with the files on their own equipment [4]. At its most complex, file and document sharing may be implemented using collaborative authoring system like a wiki. A wiki is an "interactive website" that allows users to view and edit their document content as pages [8]. All pages in a wiki are theoretically editable by any authenticated user [9]. All changes to a document build upon previous changes, allowing users to rapidly increase and enhance their content. Wiki access controls are usually based on a simple RBAC scheme, similar to forums: users acquire a *participant*

role, which allows them to edit pages in the wiki. A separate *administator* role is assigned to the wiki maintainers.

- **Document management** allows for the creation, storage, editing, sharing and publishing of centrally stored documents. A document management system often brings support for version control, templates, workflow and Role-Based access control [10]. In the simplest form, documents are downloaded from a central server and worked on locally. More complex systems such as Google Docs[1] allow users to collaboratively create, edit, share and publish documents using distributed storage. The access controls in a document management system are typically based on RBAC, although any of the three access control policies could be used.

There are a number of issues that may be identified in these collaborative systems, whether they use DAC, MAC or RBAC access control policies. Some of these issues are:

- **Costly role analysis and definition**. RBAC requires a costly initial phase to identify and define an appropriate set of roles [11]. The definition of roles and permissions can sometimes lead to what has been called "role explosion" [12].
- **Maintenance of users, objects and roles**. DAC, MAC and RBAC all require regular and on-going maintenance of users, objects and roles.
- **Low level of flexibility**. DAC policies typically implement only a coarse level of access to objects, restricting what is possible. RBAC systems are slow to adapt in fast changing environments, and may not support policies that have dynamic components such as the time of day [12].
- **Tight coupling of access control to systems**. Access control is often hard-coded into systems, limiting both their flexibility and extensibility.

As discussed above, these approaches all require on-going administration and maintenance. In the case of RBAC, a costly initial analysis is required when starting with a new system, while DAC and MAC both offer limited flexibility to their organisations. Further, tight coupling of access control to systems limits their overall extensibility.

In this paper we argue that it is time to reconsider how access controls work and propose a new paradigm based on a user's demographics and behaviour, rather than simply their identity. In this paradigm, a user's attributes and actions are evaluated and converted by policies into access permissions and roles. In our approach, access control information is derived from multiple different sources of information about a user. These sources include traditional databases or directories, but can also include non-obvious sources. Each source changes as the users it represents change, allowing it to accurately characterise the users over time. This ability to keep up with changes in user's attributes would obviate the need for regular user or object maintenance.

[1]http://docs.google.com/

Mitigating the need to hard-code access control into systems is an ability to rapidly "plug into" multiple sources of information without requiring complex adaptations or customisations. Flexibility is addressed by the ability to combine these multiple sources of information, allowing for complex and novel policies that cannot be easily captured using traditional access control systems.

Our Generalised Recommendation Architecture (GRAft[2]) provides us with a powerful, generalised recommendation system that allows us to support a range of different models. In particular, GRAft allows us to support existing reputation models, whilst also allowing new and more complex approaches. Traditional reputation systems also provide features to collect, summarise and distribute ratings about users [13], [14]. However, GRAft supports any type of recommendation, whether it is based on reputation, demographics or competencies (such as those measured and used by crowdsourcing company Amazon Mechanical Turk[3]).

The rest of the paper is organised as follows. In the next section we provide an overview of GRAft, in section III we present our case studies, and in IV our evaluation. Related work is discussed in Section V, and we conclude in Section VI.

II. An Overview of GRAft

GRAft is a federated system that supports the collection and storage of recommendations (such as reputation, competency and demographics) for entities (whether those entities are users or systems). GRAft is open, globally distributed and utilises the OpenID [15] infrastructure in order to obtain long-lived identities. All information within GRAft is distributed and duplicated over the nodes that comprise the network.

Every entity that utilises GRAft is uniquely identified using their OpenID. For each of the these entities, the system maintains a single view of their recommendations and transactional history. This information is held for each entity within a "profile". A profile is built up of multiple "profile entries". Recommendation information for an entity is derived from the transactions they conduct, but also from other sources. These sources may be either implicit or explicit sources of reputation. Our explicit reputation sources include diverse information from crowdsourcing, auction, expert and recommendation sites. Our implicit reputation sources may include bibliographic databases (for example, DBLP) and social networks.

A. Entities

1) Stack: Figure 1 represents a simple logical model of a GRAft node. This model breaks each node into a stack comprising three layers, each of which are discussed below. Profiles are obtained by the collection layer and travel up the stack to the interpretation layer.

The three layers of a GRAft node are discussed below:

- The **collection layer** is the lowest layer of the stack and is responsible for obtaining information from

[2]Graft, in the horticultural or medical rather than the pecuniary sense.
[3]http://www.mturk.com/

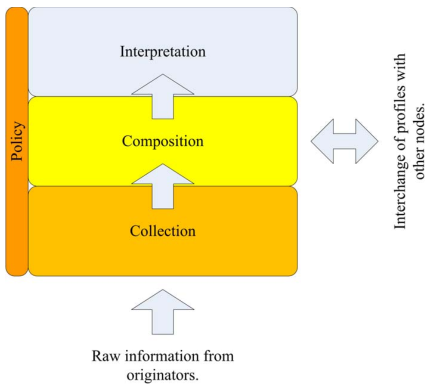
Policy
Interpretation
Composition
Collection
Interchange of profiles with other nodes.
Raw information from originators.

Fig. 1. GRAft Node Model

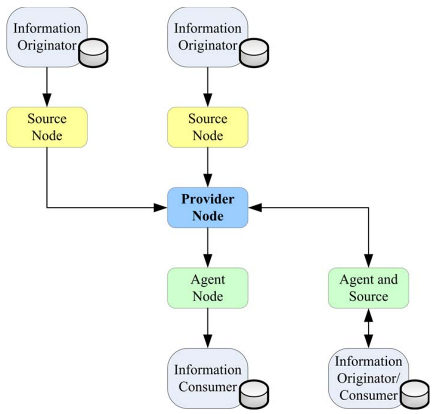

Fig. 2. GRAft Nodes

"information originators". An information originator is any source of recommendation information. All data captured by the collection layer is formatted into objects, and then passed to the composition layer. It is the job of the collection layer to obtain the information and make it available to the composition layer in a standardised form, not provide any interpretation.

- The **composition layer** is responsible for communicating with other GRAft nodes and for composing profiles received from multiple sources. In particular, this layer obtains information from either the collection layer, or from other nodes in the GRAft network. Information obtained from either source is encoded in the same way. The composition layer is responsible for organising the data obtained from these sources. This includes simplification and summarising of the information. This layer is required for a node to operate in the GRAft network, because without this layer, the node would be unable to receive profiles from other nodes or the collection layer.
- The **interpretation layer** makes decisions based on the information passed up from the composition layer. The interpretation layer is often interested in a particular entity, usually in one specific context. This layer is required to analyse profiles and make decisions based on that information.

A separate and extensible **policy layer** that forms the backbone of the three layers discussed above is also present in every node. The policy layer is responsible for interpreting, considering and aggregating any information obtained from other nodes [16]. Nodes in the network may choose to implement only a subset of the three stack layers, however every node is expected to implement the backbone policy layer. For example, an agent node that is only interested in checking the profiles of other entities may choose to implement only the composition and interpretation layers in its representative stack. A provider node is only interested in obtaining and storing profiles and may therefore only implement the collection and composition layers.

2) Node Types: The GRAft network is composed of a series of nodes with differing functions and abilities. In particular, the following three types of nodes are the most prevalent: provider, source and agent nodes.

Provider nodes are actually OpenID Providers (OPs) with GRAft extensions that allow them to store and manipulate GRAft profiles. A provider node makes available both OpenID and GRAft services, and stores information for each entity that uses the node as its OP.

Source nodes are GRAft-specific nodes that are a source of profile entries. These nodes typically only implement the bare-minimum GRAft support in order to be able to push updates to the GRAft network. The information provided by a source node is extracted from an originating application or system using either a specific API, or by accessing a database directly.

Agent nodes utilise the provider nodes to obtain information about other agent nodes (called "targets"). In particular, if a given target's provider is offline (or otherwise unavailable), the agent may retreive a copy of the profile from the GRAft network.

B. Network

All the node types in GRAft also belong to a single Kademlia DHT [17]. This DHT ensures that profiles are always available, even if the provider is currently unavailable. It also allows querying agent nodes to check the veracity of a profile shared by a provider. Figure 3 shows the different node types are all in the same Kademlia DHT.

1) Profile Verification: As there is no central control or storage within GRAft, it is useful to be able to verify that a given profile has not been modified or whitewashed. In particular, entities may implement their own provider and will therefore have full write access to their own profile. Our approach to this issue is to have a set of "super peers" that maintain a complete duplicate of each profile. The super peers are actually other nodes in the GRAft network and are accessed via the DHT.

A super peer is located by applying multiple hash functions to the OpenID of the target entity and finding the responsible nodes in our DHT. In order to overcome problems with churn, there are multiple hashes and therefore multiple copies of the profile, as nodes may leave or be unavailable at a given time. The hashing algorithm ensures that nodes are selected

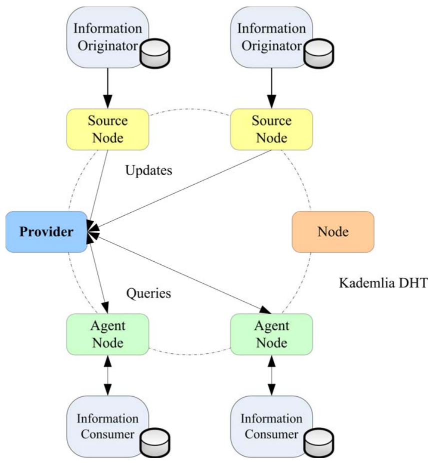

Fig. 3. GRAft Nodes with underlying DHT

in a uniform way. The amount of resource that a single node therefore needs to commit to the network is relatively low as each will only handle a small number of profiles. As the super peers are randomly allocated to profiles based on a hashing algorithm, there is little chance of duplicity. However, when checking the profile for a target entity, simple threshold voting (e.g. m of the n super peers agrees) can be used to ensure that a given profile is correct.

C. Profiles

All GRAft information is exchanged using XML objects (one per profile) that are sent and received over HTTP. Each XML object has a set of profile entries, one per unique source of information. Each profile entry contains a target, a source, a score/rating and a context. This can be used to store any valid recommendation. In each case the profile entry contains a context in addition to an actual measurement of the target's performance. Reputation is context-dependent [18], [19], and as such always requires a context in which it applies [20].

D. Operations

1) Query: An agent with a given OpenID Identifier can discover the provider for a target entity in question using standard OpenID endpoint discovery [15]. The URL discovered in an XRDS document or in the HTML document at the endpoint will point to an appropriate GRAft provider. This provider will maintain the entity's profile. Once located, a GRAft endpoint consists of a RESTful [21] web service implemented using HTTP. Each endpoint is able to fetch and store profiles, using the target entity's OpenID Identifier as a key. All complex information is wrapped in an XML container, to allow for easier parsing. Figure 4 shows how a GRAft node obtains and verifies a profile using threshold voting.

2) Update: Updates to profiles come from two sources in GRAft: agent and source nodes. In both cases profiles are fetched, modified and pushed back to the network. Figure 5 shows how a GRAft node updates a profile. We have already discussed how the profile is obtained and verified. Once verified, a profile entry is inserted or modified in the profile.

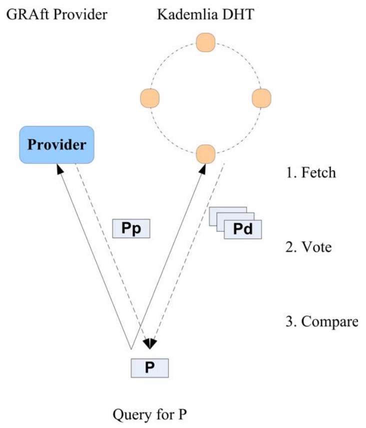

Fig. 4. Profile Query

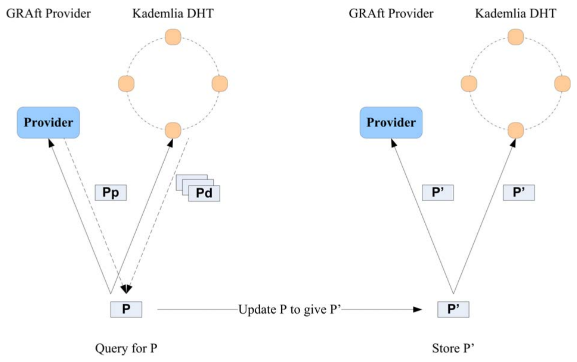

Fig. 5. Profile Update

The modified profile is then pushed back to both the provider and DHT network.

III. Case Studies

A. Forum

In this case study we are applying GRAft as the mechanism for a pure reputation system. Forum access control is enhanced by restricting users or granting them additional rights based on their reputation. In our model forum, shown in Figure 6, the reputation information for forum members in one forum is made available to a second forum. Participants in the source forum have allowed the mapping from their forum-specific identifier to their global OpenID identifier. Reputation information for those users is then "exported" via the source node to the provider node (and the network of peers). The agent node acting for the second forum is then able to retrieve reputation information for users using their OpenID identifiers. Although simple, this set up allows us to control the level of access to a forum based on a user's reputation, and further, it removes the need to bootstrap a new reputation for each additional forum, as a user is able to easily "port" their reputation from one forum to another. Context information is provided by the source node, however it is the job of the agent node to determine how useful the information in each context is, and if it should be used.

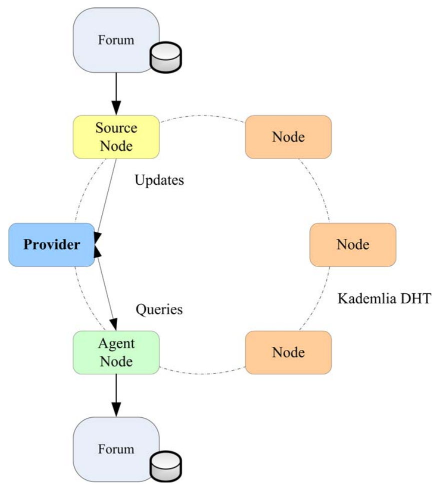

Fig. 6. GRAft enabled Forum

In our implementation, the forums are instances of the Drupal[4] framework with the userpoints, userpoints karma and forum modules installed. The source and agent nodes are implemented in PHP and have read and write access to the underlying forum databases. A policy engine implemented using Ruler[5] enforces any policies. The following policy fragments are used to restrict read access to a given sub-forum or thread, and grant the moderator role based on the reputation of the user.

```
$rb->logicalOr(
    $rb['rating']->lessThan(10),
    $rb['admin']->equalTo(null)
)

$rb->logicalAnd(
    $rb['rating']->greaterThanOrEqualTo(100),
    $rb['forum']->notEqualTo('administration')
)
```

B. Wiki

In this case study, access to wiki pages is controlled via a novel 2-tier policy. At the first tier, page access is controlled by the user's "degree of co-authorship" to the page creator. This allows the page creator to automatically grant access to people he or she has worked with previously, or more generally, those that work in the same field. The 1st degree of co-authorship is simply all those authors a given user has worked with previously. The 2nd degree of co-authorship is all the authors that the 1st degree authors have worked with, and so on. At the second tier, once the user has been granted access to the page, editorial permissions are controlled via the user's Hirsch index (H-index) [22]. A policy is responsible for assigning permissions based on the H-index value. In our case study, users could obtain read, write and moderator access to the page. DBLP[6], Microsoft Academic Search (MAS)[7], or any other sufficiently large source of citation data can be used to determine both the H-index score for the user and the "degree of co-authorship". Depending on the policy in operation, the page creator could choose to allow the cumulative total of 1st, 2nd or 3rd degree co-authors access to their page. However, it should be noted that computation times for the larger degrees increase rapidly. See the next section for more information.

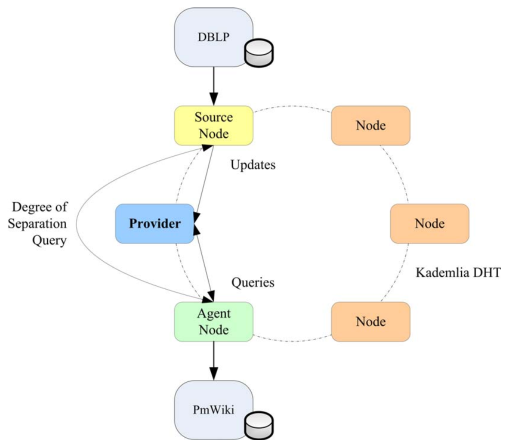

Fig. 7. GRAft enabled Wiki

Figure 7 shows the set up of this case study. In our implementation, the Wiki is an instance of the highly flexible PmWiki[8]. The agent node is integrated into the Wiki and fetches the current user's profile (and H-index) when they login. The source node is implemented in PHP and has access to an instance of the DBLP dataset (in a database). When the user tries to access a page the agent node makes a special query to the source node asking for the "degree of co-authorship" between the page creator and the current user. A policy engine controls access to the page, and also the editorial level (based on the current user's H-index). The following policy fragments show the page read and edit policies.

```
$rb->logicalAnd(
    $rb['hindex']->greaterThanOrEqualTo(1),
    $rb->logicalAnd(
        $rb['degree']->greaterThanOrEqualTo(0),
        $rb['degree']->lessThanOrEqualTo(3)
    )
)

$rb->logicalAnd(
    $rb['hindex']->greaterThanOrEqualTo(10),
    $rb->logicalAnd(
        $rb['degree']->greaterThanOrEqualTo(0),
        $rb['degree']->lessThanOrEqualTo(2)
    )
)
```

C. DataVerse

The current situation faced by scientists is one in which they are required by funding bodies and the changes to endemic scientific practice to share and preserve their data. [23] states that "Funding bodies have recently introduced a requirement that data sharing must be a consideration of all

[4]http://drupal.org/
[5]https://github.com/bobthecow/Ruler
[6]http://www.dblp.org/
[7]http://academic.research.microsoft.com/
[8]http://www.pmwiki.org/

funding applications in genomics. As with all new developments this condition has had an impact on scientific practice, particularly in the area of publishing and in the conduct of research". A little later in [24], Tenopir et al., carried out a survey of 1329 scientists across a wide range of sciences and institutions, including, medicine, biology, environmental sciences etc., and concluded that "Barriers to effective data sharing and preservation are deeply rooted in the practices and culture of the research process as well as the researchers themselves". The authors also stated that most researchers "agree they are willing to share their data" if "certain conditions are met". Significantly, scientists report "insufficient time and lack of funding" [24] as barriers to scientific data sharing and preservation.

We are currently designing the malleable access control for DataVerse [25], or a DataVerse-like scenario such as the forthcoming Globus online data sharing facility, using GRAft. DataVerse[9] is a data sharing web archive that allows users to extract, analyse, publish and reference research data. There are a number of different policies that could be applied in DataVerse, or similar scenario. For example, some of the aspects of these policies could include the "degrees of co-authorship", H-index score, affiliation, set of clients, professional associations and geographic location. Two possible policy fragments for DataVerse are shown below. These policies express the *IEEE members in the United States*, and *members of a top 500 University, who have an H-index of at least 15 and a degree of co-authorship less than 3 to the owner of the data*.

```
$rb->logicalAnd(
    $rb['member']->equalTo('IEEE'),
    $rb['country']->equalTo('United␣States')
)

$rb->logicalAnd(
    $rb['member']->equalTo('Top500'),
    $rb->logicalAnd(
        $rb['hindex']->greaterThanOrEqualTo(15),
        $rb['degree']->lessThanOrEqualTo(3)
    )
)
```

IV. Evaluation

All of our experiments are based on a single server running Debian/GNU Linux 6.0.7 with an Intel Core 2 Duo 3.0 GHz processor and 4 GB of memory. PHP 5.3.3 and Perl 5.10.1 were used.

A. Degrees of Co-Authorship

Degrees of Co-Authorship were calculated using a copy of the DBLP[10] database running on MySQL.

A sample of twenty computer science authors was selected using the number of publications they each had, ensuring that the sample included authors with both small and large numbers of publications. For each author, their publications were determined, and from that their set of unique co-authors. The set of co-authors were then used to determine their publications and co-authors, and so on for each additional degree.

[9]http://thedata.org/
[10]http://www.dblp.org/

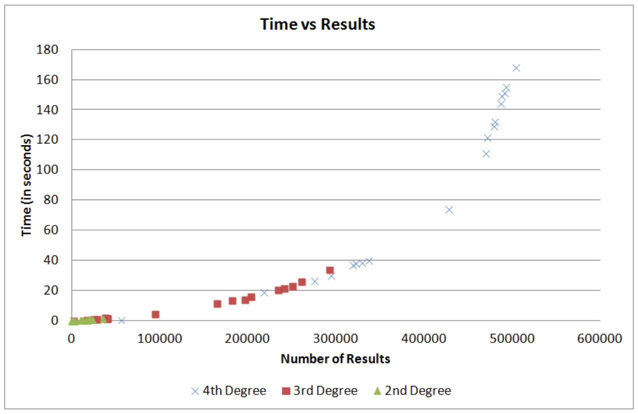

Fig. 8. Degrees of Co-authorship

Our original implementation was in PHP, however we experienced some memory allocation artefacts in our results. We therefore re-wrote our implementation in Perl. The results can be seen in Figure 8.

Only the 2nd, 3rd and 4th degree co-authorship results are plotted in this graph because the 1st degree co-authorship results are measured in thousandths of a second and are not visible at this scale. Error bars for the 95% confidence interval are also plotted, but are not visible due to their small size. Based on these results, any policies requiring more than the 2nd "degree of co-authorship" should consider using more powerful hardware, or pre-calculating the results and caching them for a defined user-set.

B. H-index Calculation

H-index values were calculated using the Microsoft Academic Search (MAS) API. This API provides well documented and easy to use SOAP and JSON interfaces. These interfaces are functionally equivalent. We elected to use the JSON interface for our implementation and evaluation because it is consistent with the RESTful approach used in our work.

Since not all authors have an author page in MAS, it was decided that our H-index values would be caculated by searching for each author's publications and sorting the results by citation count. As each response can only contain up to 100 results, multiple fetches are required for some prolific authors. The average time to fetch a page from MAS will therefore have a large impact on the time taken to calculate a H-index value.

Our evaluation therefore focused on the average time taken to fetch a page from MAS. A simple program that requested only a single result for an author search was written, effectively giving us a simple "ping" for the MAS API. This program was run every 5 minutes, and fed the name of a random author from the sample list of 20.

Figure 9 shows the average time for page requests to the MAS API for 28th April 2013 (Sunday). All requests were grouped into hourly results and plotted against the average time taken to request and retreive the result from the MAS API. The error bars show the 95% confidence interval. All times are

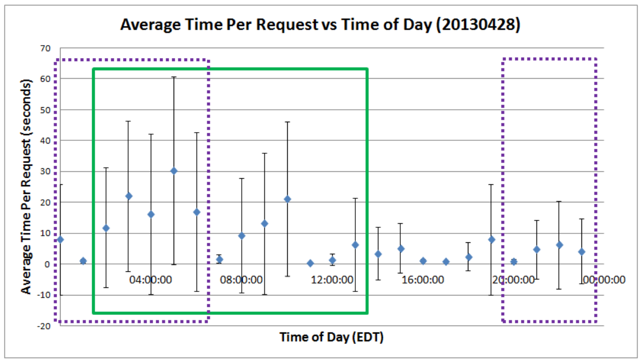

Fig. 9. MAS Average Request Time for 28th April 2013

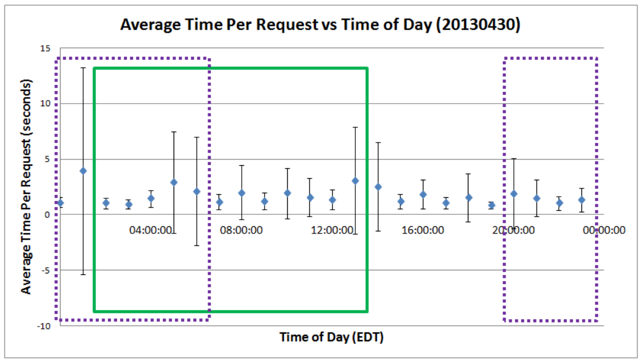

Fig. 10. MAS Average Request Time for 30th April 2013

recorded in EDT (Eastern Daylight Time). The dashed purple line shows the Chinese working day (0730 CST to 1830 CST), while the unbroken green line shows the European working day (0730 CEST to 1830 BST). The greatest variance in the response time occurs when the Chinese and European working days overlap.

Figure 10 shows the average time for page requests to the MAS API for 30th April 2013 (Tuesday). This graph exhibits much less variance than the previous graph. One possible reason for this could be the timing of the United States spring exams (approximately from 29th April through to 10th May).

V. Related Work

Slashdot[11] is a news website dedicated to technology issues. Founded in 1997, it is one of the earlier sites to employ a reputation system. All registered users have a level of "karma" that changes over time to reflect their level of engagement with the site (through the posting of articles and comments they submit). Users with good karma are able to moderate other user's comments [26]. This is similar to our work because a user's actions determine their ability to perform other tasks. However, the functionality on Slashdot is extremely limited when compared to our work, as it offers only a single source of information about a user's actions and only one additional ability.

[11] http://slashdot.org/

A more recent site that also employs this kind of functionality is Stack Overflow[12]. This website allows its users to ask and answer software development questions. Reputation points are awarded for all tasks (including asking and answering questions). As a user gains more points, they are able to access additional features on the site, including the ability to vote questions and answers up or down, and act as a moderator. Comprehensive questions or answers may be awarded further points as users with higher reputations may vote them up. This has similarities to our work, in that "good" actions lead to further abilities. However, the similarities end there, as their system is based on a simple reputation model, without the fine grained control, or the ability to use other information sources that is present in GRAft. Finally, the Stack Overflow system is contained and limited to a single silo.

In [27], Mori et al., propose a system that allows the sharing of information using social networks. Social networks are derived for every user and can then be examined. Users attach access control lists to their content, and the system decides if another user is then able to access the information. This is similar to our work, but considers only social network relationships when deciding if access is possible. Further, only a simple binary access model is supported. Later, in [28], Zhang et al., present an open social based framework for access control in eScience. They introduce and implement a user-oriented group authorization model for multi-tenanted systems and dynamic federation of community resources. Their objective, like ours, is to aid scientific data sharing, but in their case, by improving group awareness, and promoting social trust among team members within a social network.

Windley et al. in [29] and [30] introduce Pythia, a reputation-based authorization system. Reputation information is sourced from applications and stored in a central repository. Relying parties are then able to query the system and obtain reputation information about users that has been processed through a rules engine. While having similarities to our work, the work differs from our own in three keys areas: Pythia is based on a centralised architecture and is not distributed. Pythia also provides relying parties with a calculated reputation, whereas GRAft will make available the relevant information, and leave the calculation (if this is even required) to the relying party. Finally, Pythia is focussed only on reputation-based models, and does not have the ability to consider other recommendations such as a user's demographics or their competencies.

In [31], Grinshpoun et al., introduce a Cross-Community Reputation (CCR) model. They argue that reputation information sourced from multiple communities is more accurate, and removes the need to bootstrap a new reputation for every new community. The model discusses pre-conditions for sharing, reputation conversion and mapping of information from one community to another. A separate policy controls how much information is shared between the communities. The paper is similar to our work, in that reputation information can be shared amonst different communities. However, this work only talks about reputation sharing and does not provide an architecture or implementation.

[12] http://www.stackoverflow.com/

VI. Conclusion

Collaborative systems are becoming increasingly important. As with any computer system, these systems require access control to prevent unauthorised access and change. Access control has a number of issues, including administration and maintenance overheads. In this paper we have presented a new paradigm for access control. In our approach, policies transform a user's demographics and behaviour into access permissions and roles. It is a user's actions that count, not only their identity. The transformation from actions into access controls mitigates some of the constant maintenance issues seen with some access control systems, removes the need for tight coupling and adds flexibility. This flexibility allows for novel and complex policies that are currently not available in access control systems. We show through case studies how our GRAft Architecture supports and enables this new paradigm, obtaining and making recommendation information available in an open and flexible environment, establishing and facilitating scientific collaboration.

Acknowledgment

The authors would like to thank Microsoft Research for making the MAS API available for this research.

References

[1] A. M. Odlyzko, "Internet Growth: Myth and Reality, Use and Abuse," *Journal of Computer Resource Management*, vol. 102, pp. 23–27, 2001.

[2] K. G. Coffman and A. M. Odlyzko, "Internet Growth: Is there a "Moore's Law" for Data Traffic?" *Handbook of massive data sets*, vol. 142, 2001.

[3] C. S. Inc., "Cisco visual networking index: Forecast and methodology, 2010-2015," http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf, June 2011.

[4] G. Bafoutsou and G. Mentzas, "Review and Functional Classification of Collaborative Systems," *International journal of information management*, vol. 22, no. 4, pp. 281–305, 2002.

[5] P. Samarati and S. C. de Vimercati, "Access Control: Policies, Models, and Mechanisms," in *Foundations of Security Analysis and Design*. Springer, 2001, pp. 137–196.

[6] R. S. Sandhu and P. Samarati, "Access Control: Principle and Practice," *Communications Magazine, IEEE*, vol. 32, no. 9, pp. 40–48, 1994.

[7] D. Ferraiolo, J. Cugini, and D. R. Kuhn, "Role-based Access Control (RBAC): Features and Motivations," in *Proceedings of 11th Annual Computer Security Application Conference*. sn, 1995, pp. 241–48.

[8] R. Raitman, N. Augar, and W. Zhou, "Employing Wikis for Online Collaboration in the e-Learning Environment: Case study," in *Information Technology and Applications, 2005. ICITA 2005. Third International Conference on*, vol. 2. IEEE, 2005, pp. 142–146.

[9] T. R. Korsgaard and C. D. Jensen, "Reengineering the Wikipedia for Reputation," *Electronic Notes in Theoretical Computer Science*, vol. 244, no. 0, pp. 81 – 94, 2009, proceedings of the 4th International Workshop on Security and Trust Management (STM 2008). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S157106610900262X

[10] V. Balasubramanian and A. Bashian, "Document Management and Web Technologies: Alice marries the Mad Hatter," *Communications of the ACM*, vol. 41, no. 7, pp. 107–115, 1998.

[11] J. Vaidya, V. Atluri, and Q. Guo, "The Role Mining Problem: Finding a Minimal Descriptive Set of Roles," in *Proceedings of the 12th ACM symposium on Access control models and technologies*. ACM, 2007, pp. 175–184.

[12] D. R. Kuhn, E. J. Coyne, and T. R. Weil, "Adding Attributes to Role-based Access Control," *Computer*, vol. 43, no. 6, pp. 79–81, 2010.

[13] C. Dellarocas, "The Digitization of Word of Mouth: Promise and Challenges of Online Feedback Mechanisms," *Management Science*, vol. 49, pp. 1407–1424, 2003.

[14] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, "Reputation Systems," *Communications of the ACM*, vol. 43, pp. 45–48, December 2000. [Online]. Available: http://doi.acm.org/10.1145/355112.355122

[15] D. Recordon and D. Reed, "OpenID 2.0: a Platform for User-centric Identity Management," in *Proceedings of the second ACM workshop on Digital identity management*, ser. DIM '06. New York, NY, USA: ACM, 2006, pp. 11–16. [Online]. Available: http://doi.acm.org/10.1145/1179529.1179532

[16] R. Chard, "Reputation Description and Interpretation," Master's thesis, Victoria University of Wellingon, New Zealand, 2011.

[17] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System based on the XOR Metric," *Peer-to-Peer Systems*, pp. 53–65, 2002.

[18] L. Mui, M. Mohtashemi, and A. Halberstadt, "Notions of reputation in multi-agent systems: A review," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2002, pp. 280–287.

[19] Y. Wang and J. Vassileva, "Trust and Reputation Model in Peer-to-Peer Networks," in *Proceedings of the 2003 third International Conference on Peer-to-Peer Computing (P2P 2003)*. IEEE, 2003, pp. 150–157.

[20] ——, "Toward Trust and Reputation Based Web Service Selection: A Survey," *International Transactions on Systems Science and Applications*, vol. 3, no. 2, pp. 118–132, 2007.

[21] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, University of California, 2000.

[22] J. E. Hirsch, "An Index to Quantify an Individual's Scientific Research Output," *Proceedings of the National Academy of Sciences of the United states of America*, vol. 102, no. 46, p. 16569, 2005.

[23] J. Kaye, C. Heeney, N. Hawkins, J. de Vries, and P. Boddington, "Data Sharing in Genomics - Re-shaping Scientific Practice," *Nature Reviews Genetics*, vol. 10, no. 5, pp. 331–335, 2009.

[24] C. Tenopir, S. Allard, K. L. Douglass, A. U. Aydinoglu, L. Wu, E. Read, M. Manoff, and M. Frame, "Data Sharing by Scientists: Practices and Perceptions," *PLoS One*, vol. 6, no. 6, 2011.

[25] M. Crosas, "The Dataverse Network®: an Open-Source Application for Sharing, Discovering and Preserving Data," *D-Lib Magazine*, vol. 17, no. 1, p. 2, 2011.

[26] N. Poor, "Mechanisms of an Online Public Sphere: The Website Slashdot," *Journal of Computer-Mediated Communication*, vol. 10, no. 2, pp. 00–00, 2005.

[27] J. Mori, T. Sugiyama, and Y. Matsuo, "Real-world Oriented Information Sharing using Social Networks," in *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*. ACM, 2005, pp. 81–84.

[28] H. Zhang, W. Wu, and Z. Li, "Open Social based Group Access Control Framework for e-Science Data Infrastructure," in *E-Science (e-Science), 2012 IEEE 8th International Conference on*, 2012, pp. 1–8.

[29] P. J. Windley, D. Daley, B. Cutler, and K. Tew, "Using Reputation to Augment Explicit Authorization," in *Proceedings of the 2007 ACM workshop on Digital identity management*, ser. DIM '07. New York, NY, USA: ACM, 2007, pp. 72–81. [Online]. Available: http://doi.acm.org/10.1145/1314403.1314416

[30] P. J. Windley, K. Tew, and D. Daley, "A Framework for Building Reputation Systems," *WWW 2007*, pp. 8–12, 2007.

[31] T. Grinshpoun, N. Gal-Oz, A. Meisels, and E. Gudes, "CCR: A Model for Sharing Reputation Knowledge Across Virtual Communities," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, ser. WI-IAT '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 34–41. [Online]. Available: http://dx.doi.org/10.1109/WI-IAT.2009.13