Semantic Security in Service Oriented Environments

Mike Surridge, Steve J. Taylor,
E. Rowland Watkins, Thomas Leonard
IT Innovation, Southampton, UK
{ms,sjt,erw,tal}@it-innovation.soton.ac.uk

Terry Payne, Mariusz Jacyno, Ronald Ashri
University of Southampton, UK
{trp,mj04r,ra}@ecs.soton.ac.uk

Abstract

As the technical infrastructure to support Grid environments matures, attention must be focused on integrating such technical infrastructure with technologies to support more dynamic access to services, and ensuring that such access is appropriately monitored and secured. Current approaches for securing organisations thorugh conventional firewalls are insufficient; access is either enabled or disabled for a given port, whereas access to Grid services may be conditional on dynamic factors. This paper reports on the Semantic Firewall (SFW) project, which investigated a policy-based security mechanism responsible for mediating interactions with protected services given a set of dynamic access policies, which define the conditions in which access may be granted to services. The aims of the project are presented, and results and contributions described.

1 Introduction

The Grid Computing paradigm [12] facilitates access to a variety of computing and data resources distributed across geographical and organisational boundaries, thus enabling users to achieve (typically) complex and computationally intensive tasks. In attempting to realise this vision, research and development over recent years has focussed on directing Grid environments towards establishing the fundamentals of the *technical infrastructure* required, as represented by infrastructure development efforts such as the Globus toolkit [13], and standardisation efforts such as OGSA [21] and WS-Resource [9].

However, while such a technical infrastructure is necessary to provide an effective platform to support robust and secure communication, virtualisation, and resource access, other *higher-level* issues need to be addressed before we can achieve the goal of formation and operation of virtual organisations at run-time based on a dynamic selection of services [12]. In particular, whilst low-level security concerns (including encryption, authentication, etc) are addressed, the problems of describing authorised workflows (consisting of the execution of several disparate services) and the policies that are associated with service-access has largely been ignored at this level.

The enforcement of network security policies between different organisations is difficult enough for traditional computing, but becomes even more challenging in the presence of dynamically changing and unpredictable Grid communication needs. Whilst traditional static, network security policies may accommodate the types of traffic required by Grid applications, the same mechanisms can be exploited by crackers for malicious purposes, and thus security policies cannot remain static for long. Firewall policies have long been used as a mechanism to precisely determine what traffic is allowed to flow in and out of an organisational boundary, but require diligent maintenance and monitoring to ensure the integrity of the organisation and avoid breaches of security.

To permit the use of early Grid protocols, firewall policies needed to be relaxed; thus reducing control that system administrators had of their network. In addition, such relaxations could effectively leave a network prone to security breaches, and thus, as security problems arose, ports would subsequently be closed, eliminating the security problems but rendering the corresponding Grid services as useless. Thus, in many cases, some network administrators refuse to open the Grid ports in the first place.

To avoid this problem, the Grid community now tunnels many of its protocols over HTTP(S) using Web Services, exploiting the fact that these transport protocols are more widely used by other communities, and firewall policies are typically more relaxed. Of course, this just moves the target for malicious use, and may leave network administrators with even less control, as now it becomes more difficult to filter Grid traffic without cutting off other services. In effect, an "arms race" has emerged between Grid developers, malicious users and network administrators, whereby each side has to evolve and change its approach to achieve its desired function. This can result in the need to continuously update Grid applications; such problems are also found in

P2P and increasingly also in Web Service applications [19].

In this paper, we present our work on the *Semantic Firewall*, which addresses the conflict between network administrators and Grid (and other Web Service) application developers and users, by:

- analysing the types of exchanges required by typical Grid and Web Service applications[6, 5, 4];
- formulating semantically tractable expressions of the (dynamic) policy requirements for supporting these applications, including access negotiation exchanges where relevant;
- devising message-level authorisation mechanisms that can automatically determine and enforce the applicable (instantaneous) policy for each run-time message exchange;
- implementing a prototype point of control for network administrators, allowing them to manage traffic using high-level dynamic policy rules[15].

This research has been assessed against traditional Grid Application Environments such as GRIA[20] and GEMSS[8], resulting in an understanding of how dynamic security policies could work, how they could be used for security management, and possibly recommendations on the evolution of relevant Semantic Web specifications to include security, as well as prototype software.

This paper summarises the activities and research contributions of the *Semantic Firewall* project, and is structured as follows: Section 2 presents the motivation and high level issues that complicate the task of providing secure access for Grid services, and the architectural components (with justification) of the SFW are presented in Section 3. A description of how the resulting architecture implementation was evaluated is given in Section 4, followed by reflection on the lessons learned in the discussion in Section 5. The paper concludes with Section 6.

2 Motivation

It was clear from the early stages of the project that the *Semantic Firewall (SFW)* is not actually a firewall in the traditional sense (i.e. a perimeter security device), but an active component within an organisation's Grid infrastructure. Thus, the initial investigation focused on mechanisms for supporting dynamic Web Service access control, that could be deployed (and could interact) with the Web Services it was used to protect. Further protection could them be provided by traditional Firewall solutions to secure non-Grid/Web Service traffic; whilst Grid and Web Service traffic would only be routed through the specific Web Service

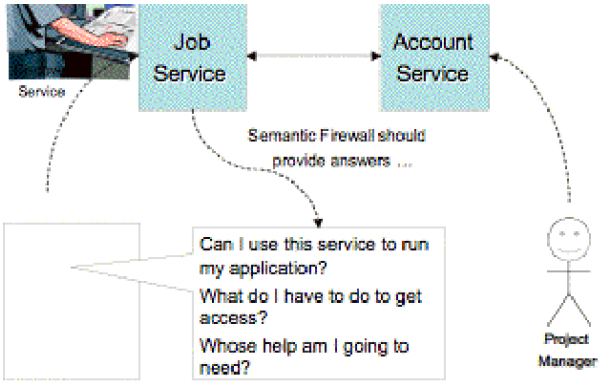

Figure 1. Discovery is just the begining

host systems mediated by the deployed Semantic Firewall components. To ensure both pragmatic acceptance and reliability, an essential feature of the SFW architecture is that network administrators have a point of control for Web Service as well as other kinds of traffic.

One of the key issues was the need in Grid applications for users to dynamically discover and exploit services. Whilst this notion works well where the user concerned has permission to use any services they can find, access is often restricted in industrial applications (at least) where their employer has to pay for services. To illustrate this, consider the scenario presented in Figure 1: an industrial Grid application user discovers a job service capable of performing computations needed by the application. However, the job service has a dynamic security policy that prevents access until the user has access to a billing mechanism. In this case, the billing mechanism is provided by an accounting service, which can only be accessed by the user's project manager. The user cannot initially access the job service; however, as the security policy is dynamic, it is possible that they could do so in future, once other actions are taken to negotiate access (i.e. induce a dynamic policy change). These dynamic situations cannot currently be addressed by simply publishing a conventional static policy (e.g. via WS-Policy[1]).

This scenario highlights many significant issues: the relationship between dynamic security and business (and possibly application) workflows, the fact that a policy may need to express the consequences of interactions between different services, and the need for communication about policy (and its possible future implications) to both users in Figure 1. Some of these issues are similar to those encountered within the Multi-Agent Systems research community[23], and suggest that the SFW should combine

[1] http://www-128.ibm.com/developerworks/library/specification/ws-polfram/

Semantic Web, agent and more conventional Web Service security technologies[5].

An emerging, but important issue to consider is that published services are increasingly being developed independently by different providers (for example, different laboratories, organisations, etc), but shared across the public domain, though service description registries, such as UDDI[2]. Whilst the appearance of Grid and web service technologies have facilitated easier access and usage of distributed services for developers, it fails to address many of the knowledge-based problems associated with the diversity of service providers, i.e. interface and data heterogeneity. Unless homogeneity can be assured though a-priori agreed interface and data model specifications, mechanisms to support interoperation between external requests, SFW policies, and the web or Grid service specifications are becoming essential.

The notion of the Semantic Web has been exploited to represent and reason about different services within open, dynamic and evolving environments[7]. The semantic web supports the utilization of many different, distributed ontologies to facilitate reasoning, as well as mappings and articulations that relate correlated concepts within different ontologies. Through the use of reasoning engines, it is possible to infer the relationships between semantically related statements (i.e. statements that are similar in meaning, if not in syntax), and thus bridge the interoperability gap between service representations, queries, and security policies defined by different sources.

Existing work on policies that are based on Semantic Web languages, provide several of the required expressive constructs for defining authorisations and obligations and their delegation. Work such as KAOS[22] takes into account some of the issues relating to conflicting policies between different domains, and provides a centralised means for resolving them. In contrast, Kagal et al. [16] assume a decentralised and adaptive model within REI, whereby the dynamic modification of policies is supported using speech acts and the suggested deployment models for this work examine different scenarios, such as FIPA-compliant agent platforms [3], web pages and web services. However, they do not take into consideration dynamic adaptation of policies within the context of particular interaction scenarios.

An additional challenge is the integration of policy languages with service discovery languages. Denker et. al. have suggested extensions to OWL-S to include policy and securitity considerations[10], but these typically relate to a single workflow, and do not consider the fact that several actors or services may be required to support dynamic, coordinated access to the type of Grid scenarios presented in Figure 1. Thus, further investigation was necessary to understand how such approaches could be extended to Grid scenarios.

[2]http://www.uddi.org/

[3]http://www.fipa.org/

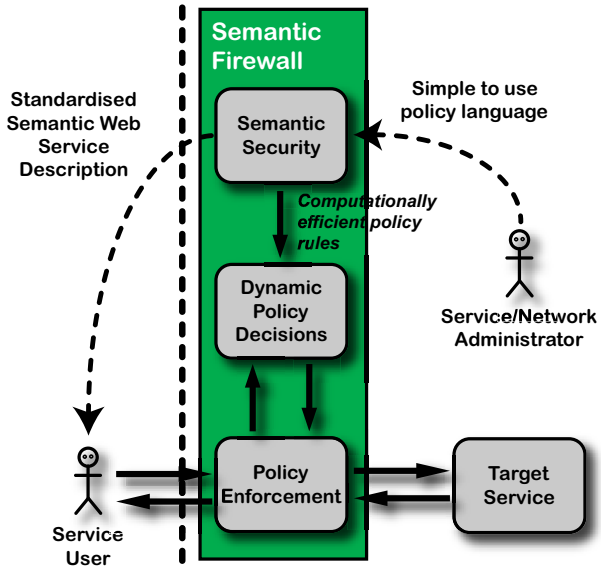

Figure 2. Semantic Firewall Architecture

3 Architecture

It is clear from Figure 1 that semantic descriptions of services will be needed, including usage processes with security constraints, so users (or their software agents) can determine whether a service can be used, and if so under what conditions. One solution is simply to represent policy constraints semantically using a policy framework such as KAOS[22] or REI[16], so they can be easily incorporated into a semantic service description[10]. However, this means that semantic inference must be used for run-time policy decisions, which leads to high run-time overheads. A consideration of the policy lifecycle phases reveals three very different requirements in the different phases (Figure 2):

- service descriptions should provide semantically tractable descriptions of behaviour, including any access policy constraints;
- policy enforcement decisions must be made rapidly without any non-trivial semantic inference, to minimise performance overheads which may delay service responses;
- access-policy must be easy to administer, independently of the application services.

3.1 Dynamic process enforcement

Two possible solutions were considered for the dynamic aspects of Semantic Firewall policy determination and enforcement:

- the SFW could monitor messages and use semantic analysis of content to infer whether access should be granted based on the current and previous message; and
- the application services could tell the SFW about state changing events, so the SFW can decide which policies to apply based on the current state.

In practice, message "sniffing" (i.e. monitoring message exchanges between service provider and client) was found to be relatively useless, because in general dynamic policy changes will depend on the logic of the hosted applications, and cannot be inferred purely from the message exchanges unless the whole application logic is reproduced in the Semantic Firewall. Instead, a finite state model definition is necessary that describes the allowed behaviour of an application (one or more related services) in a given interaction between client and service provider (these models are typically used to define the orchestration of a service using such languages as BPEL4WS[4], etc). A particular interaction is identified by a context ID, so-called because it is a reference to the prior context (circumstances occurring previously) of the interaction. Each time a user makes a request, they must quote the context ID, so the service provider knows which interaction the client is talking about. This model includes *rules* about which actions (message exchanges) each type of user can initiate in each state, and which *events* that cause state transitions. Events are messages sent by the target service itself to the SFW, so that the SFW can control which dynamic policy rules are applied to each request, as shown in Figure 3. A transactional model is then needed to specify when policy updates initiated by a service action should take effect. The SFW implements a model in which the consequences of an action on a particular context should be implemented before any other use of that context. Otherwise, race conditions may occur, in that users may be requesting actions based on out of date information.

This approach means the application service developer still encodes the control logic to determine whether a user request was "successful". However, now they must provide a state model describing the expected behaviour of their application (including "unsuccessful" and "erroneous" actions), and generate event messages corresponding to the state model. This state model describes the expected behaviour to the security infrastructure, which can thus enforce it.

[4] http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

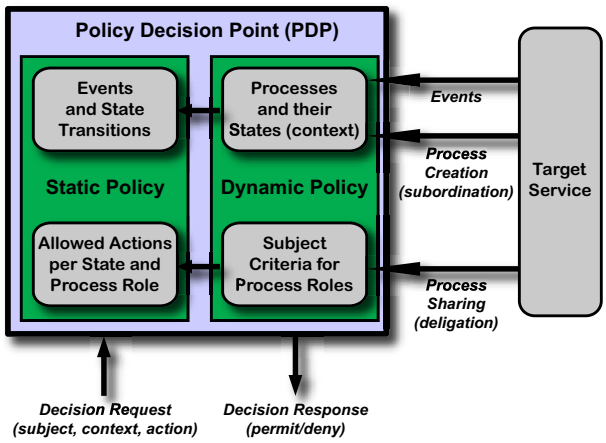

Figure 3. Dynamic Policy Component

To facilitate extensibility, and support organisational policies, the network administrator can also amend the state model, e.g. to insert additional negotiation steps or to reduce the accessible functionality. They can also integrate their own event sources into the original application, e.g. to provide a management function, a negotiation protocol, or a billing service like the account service in Figure 1. Thus, the network administrator can allow applications that require dynamic access rights for remote users, but without losing control over the possible consequences, and without being wholly dependent on the application developer to implement the desired behaviour 100% correctly.

3.2 Process roles and contexts

The permitted actions in each context depend on the state of the application service(s) in that context, and also on the type of the user (as seen by the services) in the corresponding interaction. In the SFW, user types are called *process roles*, since they are defined only with reference to service interactions and need not refer directly to a user's authenticated attributes. When an interaction process starts, a set of subject criteria are defined for each process role that relates to that interaction normally based on the attributes of the user whose action caused the new process to start. Service actions can then add or remove subject criteria for this or other process roles (a procedure known as *delegation* in the SFW), or initiate new related interactions (a procedure known as *subordination*), as shown in Figure 3.

There are often relationships between processes (contexts) and the user rights associated with them. For example, when the project manager in Figure 1 tries to open an account, they are acting in the role of "account applicant", a right delegated to them by the "service provider" when the service was originally deployed. If the service grants

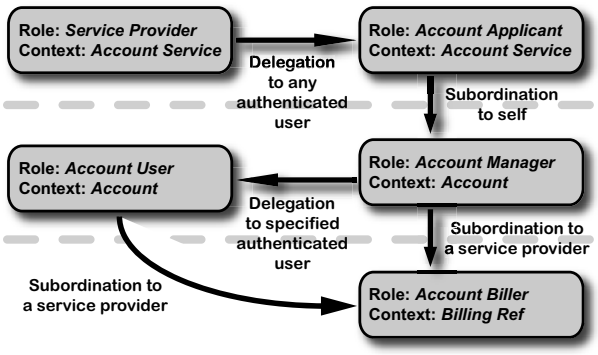

Figure 4. Process roles and contexts: delegation and subordination

the request, the account becomes a subordinate context in which the successful applicant takes the role of "account manager", and can in turn delegate charging rights (the "account user" process role) to their staff. The full hierarchy is shown in Figure 4.

The notions of process contexts and process roles, and the relationships between them, provides a way to present a high-level interface for generating security policies based on detailed state models of applications. It also provides a way to present a semantically tractable description of the overall policy, using semantic workflow languages such as OWL-S [3] to describe the externally visible processes allowed by the application state models, and RDF or OWL to describe the process roles of interacting actors, and the delegation or subordination relationships between them.

3.3 Security and the Semantic Web

The final step in the Semantic Firewall project is to develop the semantic representations of these dynamic security policies, process roles, etc. It became clear in the final year of the project that the concepts needed arise in both the Semantic Web and the Agent research communities, but they are handled in quite different ways:

Semantic Firewall	**Semantic Web**	**Agents**
Application services	WSDL	Scenes & Illocutions
Interaction protocols	OWL-S	Performative Structures
Process roles	—	Allowed Actors
Process contexts	—	—

Table 1. Applicability of Semantic Web and Agents technologies

Initial investigation suggested that Multi-Agent System mechanisms for supporting organisations, such as Electronic Institutions[11] developed at IIIA-CSIC, Spain, can be used to provide the best means to provide tractable descriptions of protected services with interaction policies. The parallels between process roles in the SFW and agent actors has been pursued through a collaboration, resulting in an analysis of the synergies and initial integration between Electronic Institutions and the Semantic Firewall[5].

A deeper analysis showed that, while the parallels are striking, the notion of a process context was far more self-contained in the Electronic Institution interaction models. After some effort to apply these principles to capture process descriptions, it became clear that significant developments would be needed to handle the idea that process contexts can be related. It was also clear that agent models may be difficult to integrate into Grid client applications, which are normally based on workflow models (for example, the Taverna system[18] from myGrid[5]). Given this, the focus moved towards semantic workflow models, and after some investigation of other options such as WSDL-S[6] and WSMO[7], the simpler but more mature OWL-S specification[3] was selected.

The main challenge was to introduce the process contexts and roles into OWL-S, and to semantically encode the permissions associated with the process roles in a given process context. It was immediately clear that OWL-S can describe a generic "process", including the notion of a "process state" that affects which actions can be taken. However, the process model as defined in the W3C submission[8] fails to support the notion that processes can be attributed to specific contexts that can be dynamically created, evolve, and be destroyed, and about which queries can be posed that should be resolvable using the generic description. As the focus for the Semantic Firewall project is to define and support dynamic security, the more general problem of such dynamism (including service discovery, provisioning, composition, etc.) were not addressed in this project. Rather, each process context was described as a distinct and separate service: for example, the transfer of data, managing offsite storage, and handling different billing accounts. Thus, in order to utilise a particular context, the service client simply uses the service description defined for a particular context.

This left only the process roles and their access rights to be added to OWL-S. This was done by defining a set of pre- and post-conditions, as follows:

1. the process state(s) from which a supported workflow could be started were encoded as pre-conditions to the corresponding OWL-S process description;

[5] http://www.mygrid.org.uk/
[6] http://www.w3.org/Submission/WSDL-S/
[7] http://www.w3.org/Submission/WSMO/
[8] http://www.w3.org/Submission/2004/07/

2. the process role(s) required to enact each workflow were also included as OWL-S pre-conditions;

3. the outcome (goal achieved) by each workflow was included as an OWL-S process post-condition;

Where appropriate, the process-role pre-conditions were accompanied by a "qualification" action, through which the actor could be granted that process role. This was encoded as a goal that would have to be achieved by finding and executing another workflow in the same context, whose outcome would be to grant the required role. Note that this second workflow also had process role pre-conditions, and typically could not be executed by the original actor. However, the pre-conditions could then be used to look for another actor (e.g. a supervisor) who could execute the required workflow.

The end result was an enrichment of the basic OWL-S process descriptions, in which the security requirements to execute each process are described, along with mechanisms that could be used (if authorised) to acquire the necessary access rights. These mechanisms typically point to other users who have control of parent contexts (for example, a prospective service user needs to acquire access permission from the manager of the account to which the service bills).

4 Validation

To validate the Semantic Firewall approach, we needed a Grid (or Web Services) application in which policies can change dynamically, e.g. as a result of a negotiation procedure. As suggested by Figure 1, such applications are typically found in inter-enterprise business Grids, such as those originally created by the GRIA[9][20] and GEMSS[10][8] projects. These Grids are forced to use at least some dynamic policy elements to handle changing business relationships, and in some cases legal constraints over the movement of data [14].

The validation study was conducted, using an alpha release of the forthcoming GRIA 5 release. This uses dynamic token services developed in the NextGRID project [1, 17] and exhibits a wider range of dynamic behaviour than the original GEMSS or GRIA systems. GRIA 5 will also use the dynamic policy decision component from the Semantic Firewall itself, which meant it was easy to integrate with a Semantic Firewall deployment for evaluation tests.

Because the GRIA 5 system is so dynamic, it was possible to test the Semantic Firewall principles even with a very simple application. The scenario involves a data centre curator transferring a large data file between two sites,

[9]GRIA is now an open source Grid middleware, currently at v4.3, obtainable via http://www.gria.org

[10]http://www.gemss.de

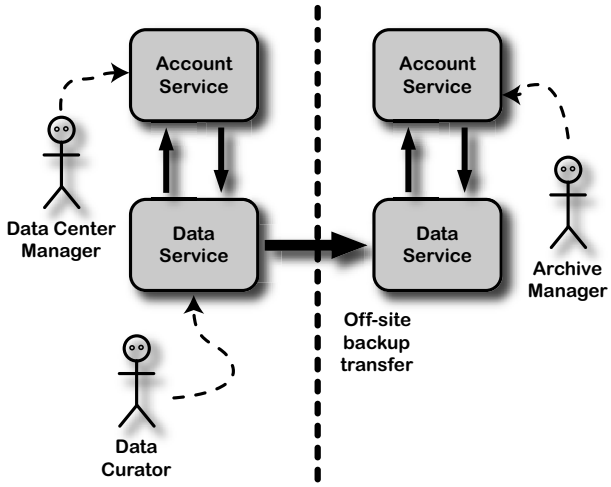

Figure 5. A simple test scenario

e.g. as part of a resilient back up strategy. The transfer is conducted using a GRIA 5 data service at each site, but these services will also bill the user for their actions, using account services located at each site. The relevant accounts are managed by other actors, e.g. the manager of the data centre where the curator works, and the manager of the network storage archive service providing the off-site backup facility. The services and actors are shown in Figure 5.

The focus of the validation exercise was on whether the SFW approach could accommodate semantically tractable service descriptions, and whether these could be used to determine if and how the curator could achieve their data transfer goal. Implementation of the application scenario was trivial using the GRIA 5 system, and registry services were added to store information about the available services (including their semantic descriptions), and information about users and their roles. Note that the user information available to each actor was incomplete the registry revealed user roles only to those actors for whom the user was willing to exercise those roles, i.e. the registry filtered according to the trust relationships between users.

To validate the use of semantic descriptions, the MindSwap OWL-S API and execution engine[11] were used to execute the required task, based on OWL-S workflows discovered from the service and workflow registry. Reasoning over the pre-conditions allowed classification of discovered services and workflows into three classes: those that could be executed by the user, those that could become executable by the user with help from other actors, and those that could never be executed. In the second case, the reasoning engine was able to work out how to negotiate the required access rights with the SFW infrastructure, with help

[11]http://www.mindswap.org/2004/owl-s/api/

from other actors (e.g. a supervisor).

In the final experiments, software agents (services) were created to represent each actor, which would take in a workflow (or goal) from other users, and (where trustworthy) enact the workflow for that user. With these services in place, it was possible to automate the negotiation process, thus exploiting the parallels and synergies between Semantic Web, agents, and dynamic security.

5 Discussion

The investigation of the Semantic Firewall to date has demonstrated that security policies can be defined for semantically described process models, with little additional run-time (semantic reasoning) overhead[15]. The research has highlighted a number of challenges for supporting dynamic, context-specific service access. The use of policies introduces a degree of flexibility for both service providers and for the network administrators. By representing these policies as state models (provided by the service providers themselves), network administrators can provide their own extension or modifications as necessary. To ensure that conflicts or violations do not occur between service and system policies, and to provide pragmatic tools to manage policy definitions, further investigation is still required.

An important consideration is that of managing policy violations, and mediating the type of response that can assist clients who are attempting to make legitimate service requests, whilst avoiding the exposure of policy details that would enable malicious users to bypass the security mechanisms. By currently exposing the defined policies, clients can reason about candidate workflows to determine whether they will fail, and thus avoid committing to workflows that would subsequently need revising. In addition, the current policy definitions allow the inference of other, necessary stages (and consequently actors) that would assist in establishing the appropriate access rights.

The decision to use a semantic representation for the SFW presented a challenge given that to associate the policies defined by the Semantic Firewall with workflows, the service description language should be extended. The OWL-S ontologies provided the ideal platform for sch extensions, and in [15] we present the set of OWL-S extensions proposed. In addition, the execution environment had to be extended to support the notion of process abstractions (through OWL-S *simple processes*) that could subsequently be provisioned at run time by consulting local service and user registries (typically within the same Virtual Organisation as the service client). For example, a client might realise that they need to acquire credentials from their local account manager before accessing those services with which such prior agreements had been made.

The research is ongoing, and future work will address extending the OWL-S extensions further, to support other notions of service. Akkermans et al. [2] introduced such notions as service bundling and the sharability and consumability of resources within the OBELIX project. Current semantic web service descriptions fail to make the distinction, for example, between resources that can be copied and shared by many users, and that which is indivisible (e.g. a security credential that can only be used by one user at the time). Likewise, there is no support for establishing relationships between service elements that support each other, but are not necessarily part of a service workflow (such as representing a billing service that supports another, primary service). Future investigation will consider how such factors augment the definition of Grid services and further support policy definitions.

6 Conclusions

In this paper, we have summarised research conducted as part of the EPSRC eScience funded *Semantic Firewall* project. The problem of providing secure access to services was analysed within a Grid scenario, which identified several challenges not normally apparent when considering simple workflows. A semantically-annotated policy model has been developed, that supports the definition of permissible actions/workflows for different actors assuming different roles for given processes. An initial prototype implementation has been developed, which has been used to validate the model on real-world, Grid case studies, and several areas for future work have been identified.

7 Acknowledgment

This research is funded by the Engineering and Physical Sciences Research Council (EPSRC) Semantic Firewall project (ref. GR/S45744/01). We would like to acknowledge contributions made by Grit Denker (SRI) for valuable insights into the requirements for semantically annotated policies and interaction workflows. Thanks are also due to Jeff Bradshaw and his research group for valuable discussions on the use and possible applicability of the KaOS framework to Grid Services, and to Carles Sierra and his group on understanding how notions of Electronic Institutions could be exploited in defining and deploying a Semantic Firewall framework.

References

[1] M. Ahsant, M. Surridge, T. Leonard, A. Krishna, and O. Mulmo. Dynamic Trust Federation in Grids. In *the 4th International Conference on Trust Management (iTrust 2006)*, Pisa, Italy, 2006.

[2] H. Akkermans, Z. Baida, J. Gordijn, N. Pena, A. Altuna, and I. Laresgoiti. Value Webs: Using Ontologies to Bundle Real-World Services. *IEEE Intelligent Systems*, 19(4):57–66, 2004.

[3] A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, D. Martin, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, and K. Sycara. DAML-S: Web Service Description for the Semantic Web. In *First International Semantic Web Conference (ISWC) Proceedings*, pages 348–363, 2002.

[4] R. Ashri, G. Denker, D. Marvin, M. Surrdige, and T. R. Payne. Semantic Web Service Interaction Protocols: An Ontological Approach. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *Int. Semantic Web Conference*, volume 3298 of *LNCS*, pages 304–319. Springer, 2004.

[5] R. Ashri, T. Payne, M. Luck, M. Surridge, C. Sierra, J. A. R. Aguilar, and P. Noriega. Using Electronic Institutions to secure Grid environments. In *10th International Workshop on Cooperative Information Agents (Submitted)*, 2006.

[6] R. Ashri, T. Payne, D. Marvin, M. Surridge, and S. Taylor. Towards a Semantic Web Security Infrastructure. In *Proceedings of Semantic Web Services 2004 Spring Symposium Series*, 2004.

[7] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):35–43, 2001.
Essay about the possibilities of the semantic web.

[8] G. Berti, S. Benkner, J. W. Fenner, J. Fingberg, G. Lonsdale, S. E. Middleton, and M. Surridge. Medical simulation services via the grid. In S. Nørager, J.-C. Healy, and Y. Paindaveine, editors, *Proceedings of 1st European HealthGRID Conference*, pages 248–259, Lyon, France, Jan. 16–17 2003. EU DG Information Society.

[9] K. Czajkowski, D. F. Ferguson, F. I, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe. The WS-Resource Framework. Technical report, The Globus Alliance, 2004.

[10] G. Denker, L. Kagal, T. Finin, M. Paolucci, and K. Sycara. Security for DAML Services: Annotation and Matchmaking. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *Proceedings of the 2nd International Semantic Web Conference*, volume 2870 of *LNCS*, pages 335–350. Springer, 2003.

[11] M. Estena. *Electronic Institutions: from specification to development*. PhD thesis, Technical University of Catalonia, 2003.

[12] I. Foster and C. Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 2003.

[13] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke. Grid Services for Distributed System Integration. *IEEE Computer*, 35(6):37–46, June 2002.

[14] J. Herveg, F. Crazzolara, S. Middleton, D. Marvin, and Y. Poullet. GEMSS: Privacy and security for a Medical Grid. In *Proceedings of HealthGRID 2004*, Clermont-Ferrand, France, 2004.

[15] M. Jacyno, E.R.Watkins, S. Taylor, T. Payne, and M. Surridge. Mediating Semantic Web Service Acces using the Semantic Firewall. In *The 5th International Semantic Web Conference (Submitted)*, 2006.

[16] L. Kagal, T. Finin, and A. Joshi. A Policy Based Approach to Security for the Semantic Web. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *2nd Int. Semantic Web Conference*, volume 2870 of *LNCS*, pages 402–418. Springer, 2003.

[17] T. Leonard, M. McArdle, M. Surridge, and M. Ahsant. Design and implementation of dynamic security components. NextGRID Project Output P5.4.5, 2006.

[18] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. C. adn K. Glover, M. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, 2004.

[19] M. Surridge and C. Upstill. Lessons for Peer-to-Peer Systems. In *Proceedings of the 3rd IEEE Conference on P2P Computing*, 2003.

[20] S. Taylor, M. Surridge, and D. Marvin. Grid Resources for Industrial Applications. In *2004 IEEE Int. Conf. on Web Services (ICWS'2004)*, 2004.

[21] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, D. Snelling, and P. Vanderbilt. Open grid services infrastructure. Technical report, Global Grid Forum, 2003.

[22] A. Uszok, J. Bradshaw, R. Jeffers, N. Suri, P. J. Hayes, M. R. Breedy, L. Bunch, M. Johnson, S. Kulkarni, and J. Lott. KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Deconfliction, and Enforcement. In *4th IEEE Int. Workshop on Policies for Distributed Systems and Networks*, pages 93–98. IEEE Computer Society, 2003.

[23] M. J. Wooldridge. *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA, 2001.