Supporting Decentralized, Security focused Dynamic Virtual Organizations across the Grid

R.O. Sinnott[1], D.W. Chadwick[3], J. Koetsier[2], O. Otenko[3], J. Watt[1], T. A. Nguyen[3]

[1]*National e-Science Centre, University of Glasgow*
[2]*National e-Science Centre, University of Edinburgh*
[3]*Information Systems Security Group, University of Kent*
r.sinnott@nesc.gla.ac.uk

Abstract

The ability to dynamically create and subsequently manage secure virtual organisations (VO) is one of the key challenges facing the Grid community. Existing approaches for establishing and managing VOs typically suffer from lack of fine grained security since they largely focus on public key infrastructures with statically defined access control lists, or they are based upon a centralised site for storage of VO specific security information. What is really needed is a federated model of security where sites are able to manage their own security information for their own institutional members, delegating where necessary to trusted local or remote entities, as well as defining and enforcing authorisation policies for their own resources. In this paper we present tools that support such capabilities and highlight how they have been applied to dynamically create and manage security focused VOs in the education domain. We believe that this federated VO security model for fine grained access to Grid services and resources should be the future model upon which security focused Grids are based.

1. Introduction

Consider the following scenario: a virtual organisation needs to be established which will make available a collection of Grid services and data sets across multiple sites to a range of users scattered across those sites. Access to these resources has strict security requirements specific to the virtual organisation being required. Sites must be able to decide for themselves whether access to their own specific resources should be allowed to VO members or not, based upon the security information that a remote user presents.

At present the establishment and management of such infrastructures is a fraught process. Understanding, agreeing upon and allocating the security attributes specific to the VO can involve the co-ordination of multiple local/remote system administrators, as well as investigators and researchers associated with the VO itself. Obtaining VO-wide agreement by all parties is also likely to require detailed negotiation.

The ability to delegate the responsibility for definition and management of the security information specific to a VO is essential for administrators at larger institutions or sites involved in numerous VOs. Whilst a site administrator might be comfortable with delegating responsibility to someone locally, the distributed nature of the Grid and multi-site VOs often requires that this delegation is made to *remote trusted* individuals. Conceptually this seems like a dangerous thing to do, however it is essential if the Grid is to scale well and be easy to manage. For example, if a remote resource provider requires specific security attributes to be provided before allowing access to their resource, then they should ideally be in control of the distribution of these attributes since they will subsequently be enforcing local authorisation decisions based upon these attributes.

To address this requires tools which build on the concept of trust. This should not be blind trust since this would be naïve and never be accepted by the Grid or non-Grid communities, but a limited and tightly controlled model of trust based upon site specific role hierarchies and delegated trust relationships.

The Dynamic Virtual Organisations for e-Science Education (DyVOSE) project [1] has developed tools that support this process building upon and extending the existing PERMIS [2] authorisation software which we describe. In section 2 we provide an overview of existing Grid security models and related work in advanced authorisation infrastructures on the Grid as well as their limitations for establishing the distributed and dynamic management of fine grained VO security. In section 3 we introduce the DyVOSE delegation model and its associated delegation issuing service (DIS) which exploits delegated trust models and controlled attribute creation and recognition to establish dynamic VOs. In section 4, we describe how the delegation model has been implemented and in Section 5 we present the educational case study that has been used as a proof of concept of the application of the DIS service. Finally in section 6 we outline the future work and exploitation plans, including how this technology could be used in combination with Shibboleth for roll out of Grid services across large scale federations.

2. Background to Grid Authorisation

The first step a Grid user has to take before he can run any Grid job is to obtain a (long lived) public key

certificate from a recognized Certification Authority (CA). Whilst this is no mean task, we don't propose to describe this procedure here, since this is the process of *authenticating* the user and this paper is concerned with *authorising* the user. Once the user has obtained his public key certificate, he must extract his globally unique Distinguished Name (DN) from it, and get the DN registered in the Grid-*mapfiles* of every Grid resource that he needs to access. A Grid-mapfile is a simplistic mechanism for authorisation, and merely maps the user's DN into a local user name known to that Grid resource. This registration will provide him with authorization to run his jobs at each resource site under the local user name mapped to him at each resource site. The Grid-mapfile lacks the ability to specify fine grained access control, or the ability to specify conditional rules for granting access, such as *only between 9am and 5pm* or *only if requested resource usage is less than a certain predefined limit*. Furthermore, it either requires lots of new local usernames to be created for all the different remote users who might run their Grid jobs locally, or if pooled accounts are used, it is unable to effectively differentiate between different remote users. Furthermore, any access control mechanism which relies solely on listing the names of authorised users is not scalable to the proportions needed for ubiquitous rollout of the Grid to existing computer users, since there are millions of them. Instead we need authorisation to be based on groups of users who share common characteristics (or attributes) such as: all members of project X, all PhD students from department Y who are IEEE members etc. This attribute based approach to authorisation is scalable to global proportions, and when it is coupled to a policy based authorisation system that allows a resource administrator to specify the rules for who should be granted which sorts of access to his resources and under what conditions, this gives us the fine grained control that we need.

We already have some procedures for assigning attributes to Grid users in the form of the Virtual Organisation Management System (VOMS) [14]. This creates a separate database for each VO, managed by the administrator of that VO. Each VO user is added to this database and given the appropriate attributes he needs to access that VO. This necessarily places a large burden on each VO administrator, since not only must he run his own separate database, he also needs to manage it and add all the Grid users to it. When we consider that most organisations have been assigning attributes to users for decades, usually in the form of LDAP attributes in organisational based LDAP directories, we are bound to ask, "why not leverage the existing attribute assigning infrastructure?" There are many possible answers to this question, two being that the organisation does not want the extra administrative burden itself, or the organisation will not give a VO manager permission to write to its LDAP server in the fear that he may disrupt the everyday operations of the organisation. In section 3 we will describe how we have built a delegation issuing service (DIS) that solves these problems and significantly reduces the administrative burden on everyone concerned, by allowing VO users to delegate permissions to each other under the prescribed limits and direct control of a *delegation policy* that is written by the organisation.

Fine grained access control may be achieved by adding a policy decision point (PDP) to each Grid resource. A PDP reads in the policy that controls access to the resource, and when asked by the application "can this user who possesses this set of attributes have this requested type of access to this resource?" it will return a granted or denied decision. PDPs have been known about and researched for over a decade. Indeed over 10 years ago the ISO access control standard [15] specified how they should work[1]. It is only recently that PDPs have been experimented with in Grid computing, thanks in part to the Global Grid Forum (GGF) which has specified a protocol for how a Grid application can call an external PDP [16]. In previous papers we have described our work in this area [4,5,8,17], in which we have plugged the PERMIS PDP into the Globus Toolkit, so that authorisation decisions can be made according to the policy specified by the resource owner. Other researchers have done similarly. Cardea [18] is a system built by NASA, that uses an XACML based PDP to make authorisation decisions for Grid jobs. The WSRF.net platform [19] at the University of Virginia also used the GGF protocol to connect a PDP to an authorisation infrastructure passing a security context and operation name. However most other Grid authorisation systems to date use Grid-mapfiles or access control lists to determine who should be granted access to the Grid resource. The limitations of access control lists are similar to those of the Grid-mapfile, in that they lack scalability, fine grained control, and the ability to specify conditions about granting access.

In the next section we will describe how we have built a system that simplifies the management of user attributes, through dynamic delegation of authority, and then uses these dynamically assigned attributes to authorise users to access Grid jobs by incorporating a PDP at the resource site.

3. Dynamic Virtual Organizations and the Delegation Issuing Service

Figure 1 shows how delegation of authority typically happens in an organisation. A resource owner (who we will subsequently refer to as the Source of Authority (SoA)) says who can use the resource in which way and signs a piece of paper to authorise this,

[1] In the ISO standard the PDP was actually called an Access control Decision Function (ADF). It was the IETF who later introduced the term PDP.

e.g. the Personnel Director signs a form to say that person A is now the head of department of the organization and can enjoy the privileges of such, or the Financial Director may say that person A can sign orders up to the value of X thousand Euros, or the computer centre manager may sign a form authorizing employee B to use a particular computing resource.

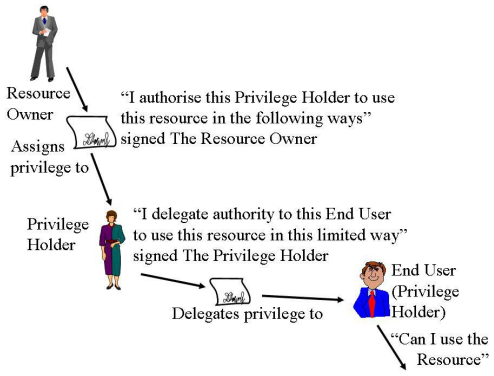

Figure 1: Delegation of Authority in an Organisation

In some situations the authorized person can further delegate (a subset of) his permissions downwards, e.g. the head of department may authorize a project leader to sign orders up to the value of Y euros (Y<X), or a user may allow another user to access one of their computer files. There are two important points to note here. Firstly the SoA will indicate if delegation of authority is possible or not, and secondly the delegated privilege must be less than or equal to the received privilege.

In VOs the delegation procedure is made more complicated because now we have multiple organisations involved. However the delegation process should remain the same, whereby people delegate to other people the ability to use the resources of the VO. Note that the delegation process we are talking about here is not the same as the usual Grid proxy-delegation process whereby a user delegates his permissions to his Grid job by assigning it a proxy certificate [23]. The primary difference is that in our model the delegate already has its own identity and means to authenticate itself, whereas in proxy delegation the grid job does not have, and the proxy certificate allows the grid job to authenticate itself as a job belonging to the user. The job then typically inherits the privileges of the user.

We have implemented the person to person delegation model in VOs as described in section 4. Because we use the role based (or attribute based) access control paradigm, the privileges that are assigned and delegated are attributes such as: member of group X, or VO work package leader. The SoA specifies in his authorisation policy who is trusted to assign and delegate which attributes to whom, and which privileges are granted to the various attribute holders. Further, each organisation in the VO specifies its own delegation policy to control the amount of delegation that may take place between its employees. A user is ultimately only granted access to VO resources according to the intersection of these two policies.

Figure 2 shows the overall delegation and authorisation model with its key components and main actors. (We have only shown two different organisations in the figure, the Service Provider (SP) and the user organisation, but the model is extensible to any number of SPs and user organisations.)

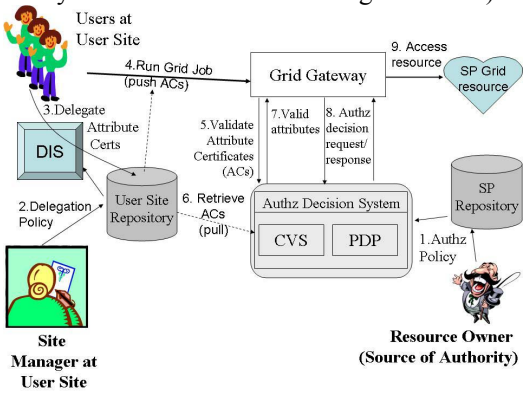

Figure 2: The Delegation/Authorisation Model

In the delegation model, the SoA for each resource writes his own authorisation policy and stores it in the local repository (step 1.) Each User Site manager writes the delegation policy for her site (step 2) and stores it in her local repository. The Delegation Issuing Service (DIS) reads in this policy at initialisation time. Users may dynamically delegate privilege attributes amongst themselves using the DIS, providing it is in accordance with the site delegation policy (step 3). When a user submits a grid job (step 4), the user's delegated and certified attributes may optionally accompany the grid job (the push mode). The grid gateway calls the authorisation decision system to validate any user attribute certificates that it received from the user (step 5), and any additional ones that the authz system may pull itself from the user site (step 6). The authz system returns to the grid gateway only valid attributes that were delegated in accordance with both the user site delegation policy and the SoA's policy (step 7). The grid gateway asks for an authorisation decision for the user's job (step 8), based on the user's validated attributes and the access control policy, and if granted, allows the job to access the grid resource (step 9)

A VO is made up of multiple resources owned by multiple SoAs. Until a SoA recognizes the existence of a VO and grants it members access rights to his resource, there is no VO from the resource's point of view. It is the SoA's authorisation policy that says who is in the VO, who is trusted to assign attributes to others throughout the VO (i.e. delegation), and which attributes grant which privileges to his resources. To facilitate the establishment of a VO each SoA must remain in full control of their resources, and need to make their own trust decisions when specifying the authorisation policy that grants access to their resource. In short, site autonomy must be upheld. If a

SoA does not trust any remote site manager to delegate the privilege attributes to users at that site, they can always directly assign them themselves. The SoA is therefore the author of the complete authorisation policy that grants access to their resource, and they can decide who to trust or not. The complete authorisation policy comprises two key components: Attribute Assignment Policy (AAP) and an Access Control Policy (ACP).

3.1 Attribute Assignment Policy (AAP)

The AAP says who is trusted to assign which attributes to which groups of users. This policy can also specify whether further delegation of these attributes is allowed or not, thus increasing scalability of attribute management. For example, a typical policy might say that site manager at site X is allowed to assign the following set of attributes to users who belong to site X, and that these users can further delegate their attributes to other site members. The most trusting policy would say that a particular manager is trusted to assign any attribute to anyone, with unlimited delegation thereafter. The least trusting policy would only trust the manager to assign one attribute to a very small group of users, with no subsequent delegation. A zero trusting policy will require the SoA to directly assign attributes to users himself. Note that this policy is under the direct control of each resource SoA, so they each only need to be as trusting as they want to be.

3.2 Access Control Policy (ACP)

The ACP says which attributes are needed in order to be granted which types of access to which Grid resource, and under what conditions. For example, a typical policy might say that members of VO *abc* (i.e. have the *abc* VO membership attribute) can retrieve certain data sets from a Grid database. A more scalable solution is to specify what attributes or roles that already exist in member organisations inherit the privileges of VO *abc* members. This eliminates the step of explicitly issuing the new attributes, thus further simplifying and increasing the scalability of attribute management.

The combined AAP/ACP policy is optionally digitally signed by the SoA (this prevents it from being tampered with) before it is stored in the local repository. The authorisation decision system reads in the SoA's authorisation policy when it is initialised by the Grid application or gateway, and will enforce this policy when users try to access the Grid resources belonging to the SoA. The Grid application or gateway can be tightly coupled to the authorisation decision engine and call it via an internal API, or can be loosely coupled and call it via the GGF SAML authorisation protocol [16] or its successors. The authorisation decision engine comprises two main components: the credential validation service (CVS) which ensures that all attributes have been issued according to the attribute assignment policy (AAP), and the PDP that ensures that all resource accesses are granted according to the access control policy (ACP).

Assuming that the SoA trusts the site manager at User Site to delegate the privilege attributes to other users at the User Site (and optionally for those users to delegate these attributes further amongst themselves – step 3) the SoA will add this site manager to his AAP.

Whatever the resource owner says becomes the policy for the resource. Note that in the traditional access control list approach the SoA needs to maintain the list of trusted people himself. Our approach allows the amount of work done by the SoA to be reduced and distributed to others, but requires the resource authorisation decision system to be more sophisticated in order to:

1. Discover the attribute assignments issued by people other than the SoA
2. Build delegation chains from these assignments
3. Authenticate the assignments as originating from a specific person or entity
4. Validate the delegation chains according to the SoA's policy; in particular, make sure that it does not end up trusting a delegated user more than it trusts the user site manager.

Considering these requirements we chose to implement the attribute assignments as a set of attributes inside X.509 Attribute Certificates (ACs) [20], because they solve the requirements as follows:

1. They can be stored in LDAP or other repositories and can be looked up by the name of the holder, the issuer or any of their contents [24].
2. Delegation chains can be built recursively by using the identities of the subjects as the next issuers in the chain. This is not possible with either SAML [21] or XACML [25], since the identities of issuers and subjects are specified differently.
3. They are digitally signed by their issuers, and public key cryptography proves authenticity of the X.509 ACs.
4. X.509 provides rules for chain validation, and by applying role (or attribute) hierarchies as well we can ensure that a delegated attribute is less than or equal to a delegator's attribute.

In addition, X.509 ACs have a compact binary encoding and usually outperform similar string encoded constructs [22]. These features are sufficient to be able to build dynamic VOs, but before looking in detail at how a dynamic VO can be constructed, we introduce the Delegation Issuing Service (DIS) that helps to optimise VO delegation of authority.

The primary drawback with the process of dynamic delegation is that delegation chains can become very long, e.g. SoA → Site Manager → Alice → Bob → Charlie etc. Thus it can become time consuming and processor intensive for the authorisation decision engine to reconstruct the

delegation chains and validate that the attributes have been delegated correctly according to the authorisation policy of the SoA. To address this we have introduced a Delegation Issuing Service (DIS) into the model to make the delegation chain processing more efficient, and in addition, to ensure that the site delegation policy is enforced at each step in the delegation process. The DIS is a service that issues X.509 attribute certificates (ACs) on behalf of the requester, according to the configured delegation policy.

The advantages of using a DIS are several. Firstly, the DIS can support a fully secure audit trail and database, so that there is an easily accessible record of every AC that has been issued and revoked throughout the organization. If each manager and user were allowed to independently issue their own ACs, then this information would be distributed throughout the organization, making it difficult or impossible to collect, being possibly badly or never recorded or even lost. Secondly, the DIS can be provided with the organization's delegation policy, and apply control procedures to ensure that a delegator does not overstep their authority by issuing greater privileges to subordinates or even to themselves, than the organization's policy allows. Thirdly, a delegator does not need to hold and maintain their own private signing key, which would be needed if the delegator were to issue and sign their own ACs. Only the DIS needs to have an AC signing key. This is a very important feature for organizations that use mechanisms other than PKIs for authentication. Once Shibboleth becomes the preferred mechanism for authentication to Grid jobs, as seems increasingly likely, then very few Grid users will have their own PKI key pairs. Finally, when the DIS is given its own AC by the site manager, it can replace the entire set of ACs in an AC delegation chain and therefore decrease the complexity of AC chain validation. The AC chain length will always be of length two when the DIS issues the ACs to end users (Site Manager → DIS AC → end user AC – see Figure 3) whereas it would be of arbitrary length when the delegators issue the ACs themselves. Also less AC revocation lists (ACRLs) will need to be issued – only the DIS will need to issue an ACRL rather than each delegator. This will further simplify AC chain validation. Note that a full audit trail of the logical delegation chain is still available, since the DIS adds an "issued on behalf of" field into all its issued ACs, in accordance with the 2005 edition of X.509.

4. DIS Implementation

The DIS is a web service that can be called by any other web service. In order to provide a user friendly web browser interface, it has been deployed with an Apache Axis Web Service front-end written in PHP which forms a proxy between the user (web browser) and the DIS service itself. Mutual SSL authentication takes place between the Apache server (acting as the DIS client) and the DIS web service, and the Apache DIS client invokes the DIS Java library through SOAP calls. This allows the Apache DIS client and the Tomcat-served DIS server to be hosted on separate resources if required, although for the purposes of our explorations they were mounted on the same machine.

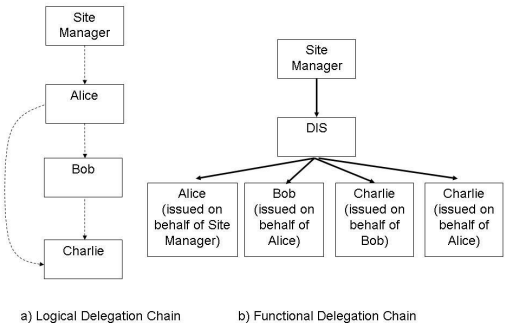

Figure 3: Delegation Chains with the DIS

The user may authenticate to the Apache server in whatever method a site prefers. In our experiments we used the Apache authentication module *mod_auth_ldap* in which usernames and passwords are stored in a backend LDAP server. The Apache DIS client is a trusted proxy of the DIS server, consequently the DIS server will issue ACs on behalf of the authenticated user, providing the user has sufficient permissions according to the site delegation policy.

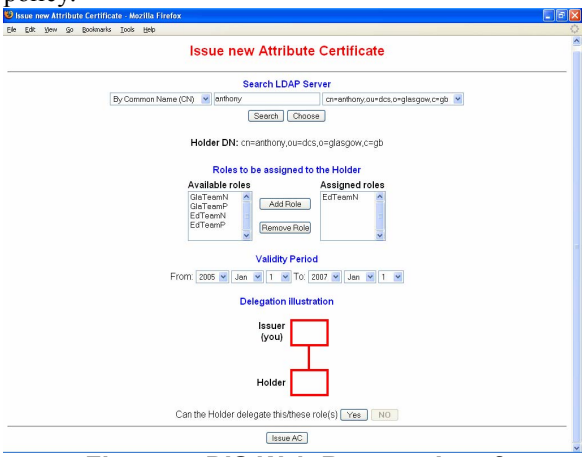

Figure 4: DIS Web Browser Interface

When using the DIS, an authenticated user may search the local LDAP for the user they wish to delegate an attribute to, and then pick the attribute from the presented picking list, as shown in Figure 4. The user then sets a validity date for the delegation, and finally grants a level of further delegating ability to the delegate as shown in Figure 4. In Figure 4 a Glasgow user (CN=Anthony, OU=DCS, O=Glasgow, C=GB) is being delegated a VO role (*EdTeamN*). Upon submission of this request to the DIS, the local delegation policy is checked to make sure the delegator (the person who logged into Apache – the delegator's ID is not shown on the above web page) is allowed to delegate the *EdTeamN* attribute to

Anthony, and then the AC is either issued or an error message is reported.

Previous PERMIS tools allowed any user to issue any AC to anyone, with validity checking being carried out by the authorisation system at the resource site. The latter step must still be carried out at the resource site, even when a DIS is being used, since a user may have a valid credential at his home site that is not trusted at a remote resource site. However, the DIS allows more control to be exerted over the delegation process, and gives the delegator immediate feedback as to whether his delegation is authorised or not by his home site.

The web browser based front end to the DIS allows AC issuing and delegation to be performed anywhere in the world by anyone who can authenticate to the Apache server. So if a researcher is away from his laboratory and wants to authorise a colleague to run an experiment on his behalf, he can issue an AC to his colleague via the DIS, delegating him the appropriate attributes for the appropriate duration of his absence. It also allows VO attribute management to be delegated to other administrators in the PMI with the minimum of difficulty. For example, consider a VO where it has been agreed amongst the institutions that access to some resources is dependent on some low-privilege attribute called 'BasicUse', which through the PERMIS authorisation policy will allow access to less resources than, say, the "Employee" attribute. A resource provider who wishes to participate in the VO may offer his resource to VO users provided they hold the 'BasicUse' attribute. In a scenario such as this, users need to be given the "BasicUse" attribute to present to a resource site. This can be achieved in several ways. Either a resource owner (SoA) can trust the user site administrator to delegate the "BasicUse" attribute to her users, by including her as a trusted AA in his PERMIS authorisation policy, or a user site administrator may trust the SoA to assign the 'BasicUse' attribute to her users. In the former trust model, the administrator may choose to issue the "BasicUse" attributes herself, or empower the users to dynamically do this using the DIS. In the latter trust model, the user institution must create a new user ID for the SoA, give him authentication credentials to the DIS Apache client, and then allow him to delegate the "BasicUse" attribute to their local users. A third approach, which relies on the hierarchical RBAC model, is for the SoA to define the 'BasicUse' attribute as subordinate to one or more existing attributes e.g. "Employee" that have already been allocated to users. In this case the users will inherit the privileges to access the SoA's resource through possession of their existing attribute(s).

In all of these scenarios the precept that sensitive user information should be held at the user's institution is still valid. The attribute(s) that are more relevant for accessing external resources may be safely allocated by either the local administrator (who has ultimate authority at the user site) or by the external resource provider (who is allocating an attribute useful for accessing his resource). In both cases the attributes are still stored at the user's home institution. If the number of users to be assigned 'BasicUse' is large, then the local administrator may wish to allow the remote SoA to allocate this attribute to save himself work, safe in the knowledge that the remote SoA can only ever act within the bounds of the delegation policy that he has set.

Note that with the third approach, role-mapping between institutions must be agreed before this kind of inherited authorisation can take place, as the "Employee" attribute may have different names in each institution of the VO. PERMIS software that allows multiple attribute mappings is currently being developed, but for the DIS case study, the attributes were simply given the same name at each institution. The scenario described above has been implemented during the second year of the DyVOSE project and is described in the next section.

The software prerequisites for implementation of the DIS are relatively straightforward to configure and interface. However in order to realise the secure issuing of ACs at a user site that are deemed to be valid at a SP site, a unified underlying PKI is required. This is so that AC signatures created at one site can be validated at another site. The DIS service ships with an example working configuration that assumes a unified PKI and a single root CA, but porting the DIS to support an established local independent PKI security infrastructure is a non-trivial step that requires careful public key certificate management. Detailed information on how multiple PKIs can be supported is given in [1].

5. Case Study

The National e-Science Centre (NeSC) at the University of Glasgow teaches Grid computing to advanced MSc students. This course has been run for the last two years. In the first year, the student programming assignment (developing secure Grid services to search and sort the complete works of Shakespeare) was used to (successfully!) show how a static PMI could be defined and used based upon technology such as Globus toolkit version 3.3 [3], Condor (www.cs.wisc.edu/condor) and PERMIS [2]. The experiences and results of this activity are presented in [6, 7, 9, 11].

The programming assignment associated with the second year of teaching Grid Computing focused upon the *dynamic* establishment and usage of a PMI in a bioinformatics related project. The case study separated the eleven students that took the Grid computing module into two different research teams: one team analyzing protein sequence data and the other analyzing nucleotide sequence data. The students were required to implement a Globus-based Basic Local Alignment Search Tool (BLAST) bioinformatics application to perform the analysis,

which was to run across a Condor pool located at NeSC in Glasgow. Before they could perform the analysis, the students were expected to develop a Grid client to retrieve the data from a PERMIS protected data service (*BlastData*) in Edinburgh. Depending upon the *team* that they were in, either protein or nucleotide sequence data would be returned respectively. These data sets were then used as input to a Globus GT3.3 based Grid service which the students implemented locally at Glasgow. These services parallelized the BLAST application over the NeSC Condor pool. Diagrammatically the assignment and associated infrastructure is given in Figure 5.

We note that for simplicity, the genomic databases used to BLAST against were pre-deployed across the Condor pool. The students Grid BLAST services themselves were protected through PERMIS so that only members of the appropriate student team were able to invoke their respective BLAST services. The two different authorization policies were pre-defined and deployed in the local NeSC LDAP server by the teaching staff. The students were able to secure their Grid services through including the appropriate authorization policy in their deployment descriptor file as described in [5].

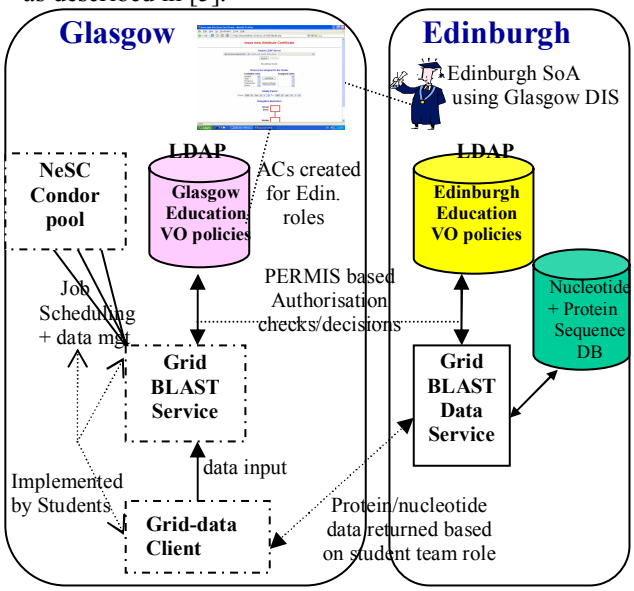

Figure 5: Dynamic PMI Case Study Infrastructure

In the initial implementation the *Edinburgh SoA* used the Glasgow DIS service to issue attributes within the Glasgow PMI for the two roles (*EdTeamP* or *EdTeamP*) needed to gain access to the Edinburgh-based *BlastData* Grid service, i.e. the SoA was delegated the privilege by the *Glasgow Administrator* to assign subordinate roles within the Glasgow role hierarchy. In this scenario the *Glasgow Administrator* delegated the privilege to the *Edinburgh SoA* to issue attribute certificates to roles below *externalStudent*.

In the above model the service provider (here the Edinburgh *BlastData* provider) is able to select via the DIS which users (from Glasgow) should be allocated the specific role needed to access and use his service. In this model, the Edinburgh service provider has fine grained control of the attributes and specific users who should be given them. For smaller tightly controlled VOs, this model might be beneficial. For larger scale VOs however it might be better for the service provider to simply delegate this responsibility to an appropriate remote administrator, e.g. in this scenario the Grid computing course director at Glasgow might be delegated the privilege to issue *EdTeamN/EdTeamP* roles to students at Glasgow. Both of these scenarios are fully supported by the DIS and have been explored in DyVOSE.

Through creation of a VO specific role within the Glasgow role hierarchy by the *Edinburgh SoA*, Glasgow students were subsequently able to access and return the appropriate sequence data sets for input to the BLAST service. To understand how this was achieved, we consider the basic interactions that take place when a Glasgow student Grid client attempts to retrieve data from the Edinburgh Grid *BlastData* service. The user certificates that were issued to all students undertaking the Grid Computing course were from a local certification authority established at NeSC Glasgow for teaching purposes. It is based on openSSL. These certificates were used to generate proxy certificates used by the Grid client through the Globus Security Infrastructure (GSI) [13] in order to interact with the secure Edinburgh *BlastData* service. Based upon the authorization system associated with this service, the associated Edinburgh PERMIS policy is queried to make an authorization decision. This policy includes information that states that a remote repository exists (the Glasgow LDAP) server, where user ACs may be found. The PERMIS CVS then concurrently queries both its local repository and the remote Glasgow LDAP server to pick up the ACs of the current user. In this case, the Glasgow LDAP identifies that an AC exists and returns it to the Edinburgh CVS, in order for it to validate the delegation chains. The Edinburgh CVS checks the signature associated with this AC and once satisfied it comes from a trusted source, the attribute itself is checked (*EdTeamN/EdTeamP*) by the PDP and the appropriate data set returned. Note that if the attribute associated with the Glasgow student does not match the one required in the access control policy, e.g. the student was in the nucleotide team and was attempting to access protein data then the access request will be denied.

Of the 11 students that undertook the Grid Computing module and associated programming assignment this year, several successfully completed all of the assignment. All of the students that built the Grid client were able to return data from Edinburgh. As such, this education domain application of the DIS has shown the proof of concept for how a service provider (*BlastData*) is able to either securely create or make use of dynamically issued attribute certificates at remote sites which subsequently allow

for fine grained authorization decisions on their resource. It is our intention to explore this service across a wider range of other projects in numerous other domains where fine grained security is needed [10, 12].

6. Conclusions

The models and implementations presented in this paper have the potential to revolutionise the way Grids are built and particularly the way in which dynamic VOs are established and managed. Decentralised models are a better and more scalable model for VOs than centralised VO models, and more aligned with the true vision of the Grid in supporting dynamic collaborations and resource sharing.

The formation of any VOs depends upon trust. In this paper we build upon a basic model of trust and show how controlled delegation of authority allows for rapid VO formation and fine grained authorisation. Through the DIS we have shown how service providers can delegate authority to remote trusted administrators to dynamically issue roles/attributes needed for access to/usage of their services. Other models where the remote service providers themselves are trusted by local administrators to issue roles/attributes to local users have also been explored with the DIS. Both models have their own advantages and disadvantages depending on the VO requirements and the levels of trust between the parties. Time will tell which approach to using the DIS has most take-up by the Grid community.

We note that the DIS service is also highly complementary to Shibboleth which is being rolled out across UK academia. Shibboleth assumes a core, largely static set of attributes which identity providers may release to service providers to make authorisation decisions for access to protected resources. Pulling VO specific attributes dynamically created through DIS via Shibboleth will harmonise the access and usage of Grid and non-Grid resources.

6.1 Acknowledgements

The DyVOSE project was funded by a grant from the Joint Information Systems Committee (JISC) as part of the Core Middleware Technology Programme.

7. References

[1] Dynamic Virtual Organisations for e-Science Education (DyVOSE) project, www.nesc.ac.uk/hub/projects/dyvose/
[2] D.W.Chadwick, A. Otenko, E.Ball, Role-based Access Control with X.509 Attribute Certificates, IEEE Internet Computing, March-April 2003.
[3] Globus toolkit, www.globus.org/toolkit
[4] R.O. Sinnott, D.W. Chadwick, Experiences of Using the GGF SAML AuthZ Interface, Proceedings of UK e-Science All Hands Meeting, September 2004, Nottingham, England.
[5] R.O. Sinnott, A.J. Stell, D.W. Chadwick, O.Otenko, Experiences of Applying Advanced Grid Authorisation Infrastructures, Proceedings of European Grid Conference (EGC), LNCS 3470, pages 265-275, Volume editors: P.M.A. Sloot, A.G. Hoekstra, T. Priol, A. Reinefeld, M. Bubak, June 2005, Amsterdam, Holland.
[6] R.O. Sinnott, A.J. Stell, J. Watt, Experiences in Teaching Grid Computing to Advanced Level Students, Proceedings of CLAG+Grid Edu Conference, May 2005, Cardiff, Wales.
[7] A.J. Stell, R.O. Sinnott, J. Watt, Comparison of Advanced Authorisation Infrastructures for Grid Computing, Proceedings of International Conference on High Performance Computing Systems and Applications, May 2005, Guelph, Canada.
[8] J. Watt, R.O. Sinnott, A.J. Stell, Dynamic Privilege Management Infrastructures Utilising Secure Attribute Exchange, Proceedings of UK e-Science All Hands Meeting, September 2005, Nottingham, England.
[9] R.O. Sinnott, J. Watt, O. Ajayi, J. Jiang, J. Koetsier, A Shibboleth-Protected Privilege Management Infrastructure for e-Science Education, 6th IEEE International Symposium on Cluster Computing and the Grid, CCGrid2006, May 2006, Singapore.
[10] R.O. Sinnott, A.J. Stell, O. Ajayi, Development of Grid Frameworks for Clinical Trials and Epidemiological Studies, HealthGrid 2006 conference, Valencia, Spain, June 2006.
[11] R.O. Sinnott, O. Ajayi, A.J. Stell, J. Watt, J. Jiang, J. Koetsier, Single-Sign on and Authorization for Dynamic Virtual Organizations, International Conference on Virtual Enterprises, (PRO-VE'06), Helsinki, Sept. 2006.
[12] Meeting the Design Challenges of nanoCMOS Electronics, EPSRC pilot project to begin October 2006.
[13] Globus Grid Security Infrastructure (GSI), http://www-unix.globus.org/toolkit/docs/3.2/gsi/
[14] Alfieri R, et al. VOMS: an authorization system for virtual organizations, 1st European across Grids conference, Santiago de Compostela.
[15] ITU-T Rec X.812 (1995) | ISO/IEC 10181-3:1996, Security Frameworks for open systems: Access control framework.
[16] Von Welch, Rachana Ananthakrishnan, Frank Siebenlist, David Chadwick, Sam Meder, Laura Pearlman. "Use of SAML for OGSI Authorization", Aug 2005.
[17] D. W Chadwick, O. Otenko, V. Welch, Using SAML to link the GLOBUS toolkit to the PERMIS authorisation infrastructure, Proceedings of 8th Annual IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, Windermere, UK, September 2004, pp251-261
[18] Lepro, R., Cardea: Dynamic Access Control in Distributed Systems, NASA Technical Report NAS-03-020, November 2003
[19] Web Service Resource Framework .NET, http://www.cs.virginia.edu/~gsw2c/wsrf.net.html
[20] ISO 9594-8/ITU-T Rec. X.509 (2001) The Directory: Public-key and attribute certificate frameworks
[21] OASIS. "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard, 15 March 2005
[22] D. Mundy, D.W. Chadwick, An XML Alternative for Performance and Security: ASN.1, IEEE IT Professional, Vol 6., No.1, Jan 2004, pp30-36
[23] S. Tuecke, V. Welch, D. Engert, L. Pearlman, M. Thompson, Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile, RFC3820, June 2004.
[24] S. Legg. "Lightweight Directory Access Protocol (LDAP) and X.500 Component Matching Rules". RFC 3687, February 2004.
[25] OASIS "eXtensible Access Control Markup Language (XACML) Version 2.0" OASIS Standard, 1 Feb 2005