GridShib and PERMIS Integration

Chadwick, D.W., Novikov, A., Otenko, O., University of Kent, United Kingdom

Abstract

This paper describes the results of our recent GridShibPERMIS project to provide policy-driven role-based access control decision making to Grid jobs, in which the user's attributes are provided by a Shibboleth Identity Provider (IdP). The goal of the project is to integrate the identity-federation and attribute-assignment functions of Shibboleth with the policy-based enforcement function of PERMIS, in order to provide a flexible fine-grained authorisation system for Grid jobs running under Globus Toolkit v4. This was done by taking the GT4-Shibboleth integration performed in the United States with the PERMIS infrastructure built in the United Kingdom, and developing a GridShibPERMIS Context Handler. This allows for interoperability between GridShib and PERMIS by providing the required attribute extraction, conversion and transfer functions. As a result, the GridShibPERMIS project integrates the advantages of both Shibboleth cross-organisation identity federation and PERMIS policy-driven role-based access control and represents a new avenue of policy-based authorisation for Grids. The paper provides a brief overview of the technologies involved: GT4, Shibboleth and PERMIS, and presents how the three are combined to provide an efficient and simple fine-grained authorisation mechanism, having low implementation costs. The paper concludes with the lessons learned and plans for the future.

Key words

Grid computing, Privilege Management Infrastructure, Authorization, RBAC, Attribute Certificates, PERMIS, Shibboleth, Globus Toolkit.

1. Introduction

Current progress in Grid computing (Foster and Kesselman, 2004) opens up new opportunities for distributed multi-institutional collaborations through Virtual Organisations (VOs) (Foster et al., 2001). Grid computing dramatically increases the ability of academic institutions to accumulate and analyse data for research purposes, but, in turn, raises issues of secure authentication and authorisation. At the moment, in the existing Grid solutions, such as Globus Toolkit (Foster, 2005),

security is usually based on the identities of the interacting parties. However, with the expansion of VOs this approach seems to lack scalability and flexibility, as it is unable to cope with dynamically changing users, rights and permissions. It also raises privacy concerns due to the release of personal information for user authorisation.

The outlined problems can be solved by expressing user's rights as attributes or roles rather than based on the user's identity. This will allow for fine-grained access control and the necessary level of confidentiality. At the moment, a number of projects provide an attribute-based authorisation framework, including Shibboleth (Morgan et al., 2004), PERMIS (Chadwick and Otenko, 2003), Akenti (Thompson et al., 1999) and VOMS (Alfieri et al., 2003). However, these competing solutions are often based on incompatible standards and require users to have different authorisation tokens for interoperability with Grids. Although there are continuous attempts to develop a common authorisation framework for the Globus Toolkit, significant effort is still required to provide an integrated solution.

In this paper, we describe our work to integrate the combined Globus Toolkit and Shibboleth infrastructure (GridShib) with the PERMIS authorisation decision engine, in order to provide policy-driven role-based access control to Grids. The goal of our project is to integrate the identity-federation and attribute-assignment functions of Shibboleth with the policy-based enforcement function of PERMIS, in order to authorise Grid jobs running with the Globus Toolkit v4. Shibboleth is an inter-realm authentication and attribute-based access control solution already being experimented with in a large number of research and educational institutions. Using it for authorising Grid jobs will allow us to leverage its existing security infrastructure, in particular, the assignment of attributes to users. The addition of the policy-driven RBAC features of PERMIS will provide the necessary level of fine-grained access control and flexibility in assigning and changing users' permissions.

This paper starts with a brief description of the Globus Toolkit (section 2), a description of the PERMIS infrastructure (section 3), the Shibboleth infrastructure (section 4), and an overview of the GridShib research project (section 5). Section 5 also presents the limitations of the current GridShib solution from an authorisation perspective. Section 6 describes our work on integrating GridShib and PERMIS. Section 7 concludes with the lessons learnt and directions for the future.

2. Globus Toolkit

The open-source middleware Globus Toolkit, provides all the necessary functionality for running Grid jobs and supports the development of service-oriented distributed applications and infrastructures. The Globus Toolkit is designed to enable applications that federate Grid resources such as computers, data, services and networks, and is already widely adopted as a means of building large-scale academic and commercial distributed applications. The recent version four of the Globus Toolkit has seen the convergence of Grid and Web services through the adoption of the standards of the Web Services Resource Framework.

In order to address the security issues arising from the sharing and coordinated use of resources in distributed VOs, Globus Toolkit provides the Grid Security Infrastructure (GSI) (Welch et al., 2003), which is based on public key cryptography and includes transport-level and message-level security mechanisms (Welch, 2005a). GSI authentication and authorisation capabilities are built upon the use of X.509 end-entity certificates (Welch et al., 2004a) and X.509 proxy certificates (Tuecke, 2004), which assert the user's identity expressed as a unique X.500 Distinguished Name (DN). Proxy certificates allow a user with a valid X.509 certificate to temporarily delegate his identity (and privileges) to another entity – his Grid job – by issuing it with a proxy certificate, thus easing the authentication (and authorisation) process. Since the DN of a proxy certificate must be subordinate to the DN of the user, the Grid job inherits some or all of the user's privileges. Because the Grid job is given its own PKI key pair and proxy certificate, it may authenticate to various systems on the Grid without additional input from the user. This provides single sign-on support for users of Grid jobs and their spawned tasks. The spawning of tasks is enabled as these are issued with their own proxy certificates by the Grid job, without any further input from the user.

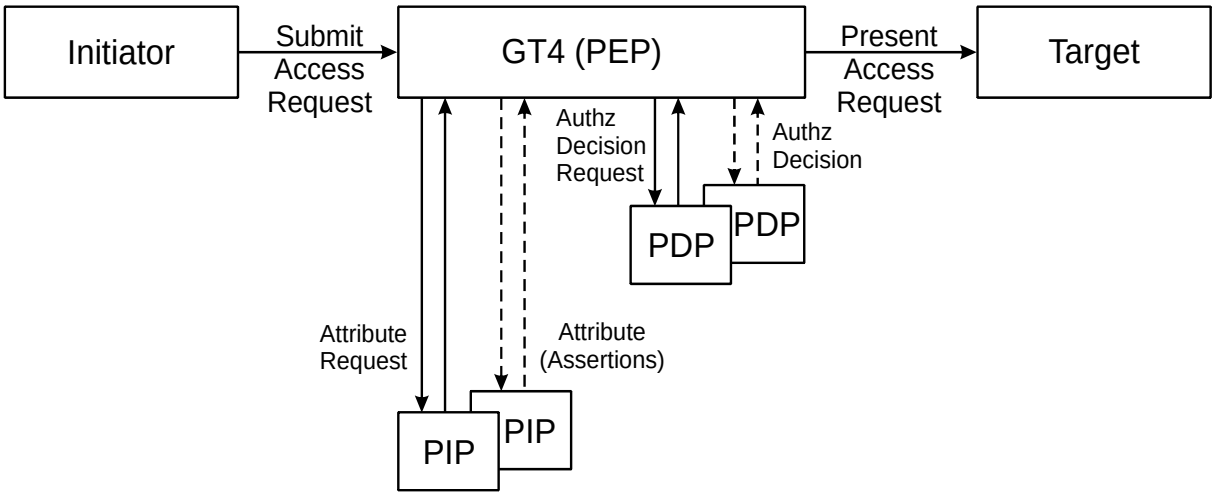

Figure 1: GT4 Authorisation Architecture

The GSI server-side authorisation framework implements the decision engine, which evaluates a chain of authorisation schemes – Policy Decision Points (PDPs) – in order to determine the access rights of the user making a request for a particular Grid service or resource (the target in Figure 1). This authorisation chain may also include Policy Information Points (PIPs), which do not return any decisions but instead are used to collect information i.e., attributes or attribute assertions, necessary for the decision-making process. Both PDPs and PIPs are classified by the Globus Toolkit as interceptors. The Globus Toolkit itself is the Policy Enforcement Point (PEP) that enforces the decisions made by the PDPs, and passes the information returned by the PIPs to the PDPs. However, authorisation in GSI is, by default, based on access control lists (ACLs) located in a *grid-mapfile*, which specifies the DNs of the users allowed to access each Grid resource. This approach provides very limited functionality and is inconvenient for large-scale distributed systems as it lacks the necessary scalability and flexibility in describing users' rights. A user is essentially granted all rights or no rights, depending upon whether his DN is in the *grid-mapfile* or not.

Starting with version 4, the Globus Toolkit (GT4) GSI uses the SAML standard (SAML Specification, 2005) from OASIS to access external third party authorisation-decision services, through either the SAML *AuthorizationDecision* protocol callout (Welch et al., 2004b) to PDPs or the SAML *Attribute Request* protocol callout to PIPs, thus allowing specialised PDPs and PIPs to be included in the GT4 authorisation chain. These callouts are used in this project to carry out the integration and enhance the authorisation options of Globus Toolkit through the use of Shibboleth as the PIP and PERMIS as the PDP.

3. PERMIS

PERMIS is a policy-based authorisation system which uses policies written in XML to support the role-based access control (RBAC) paradigm (Chadwick, 2001). Given a user's DN, a resource and an action, PERMIS says whether the user is granted or denied access based on the RBAC policy for the resource and the user's validated roles and attributes. A core component of PERMIS is to provide the PDP functionality shown in Figure 1. However, it provides more than this, as shown in Figure 2.

PERMIS is based on the X.509 Privilege Management Infrastructure (PMI) (ISO, 2001), which uses Public Key Infrastructure principles of operation. It introduces the X.509 attribute certificate

(AC) as the credential which maintains a strong binding between the user's unique DN and one or more of his privilege attributes, much the same way as a public key certificate binds the user's DN to his public key. An AC is signed by the Attribute Authority (AA) that issued it, whilst the root of trust for the PMI target resource is called the Source of Authority (SOA). An X.509 PMI can support the discretionary access control model by storing users' access rights in their ACs and by allowing users to issue ACs for the resources they control. Alternatively, an X.509 PMI can support the RBAC model by storing users' roles and attributes in ACs and allowing managers to issue the attributes and roles they control. Resource owners may then publish ACs detailing the access rights that they have granted to each role or attribute.

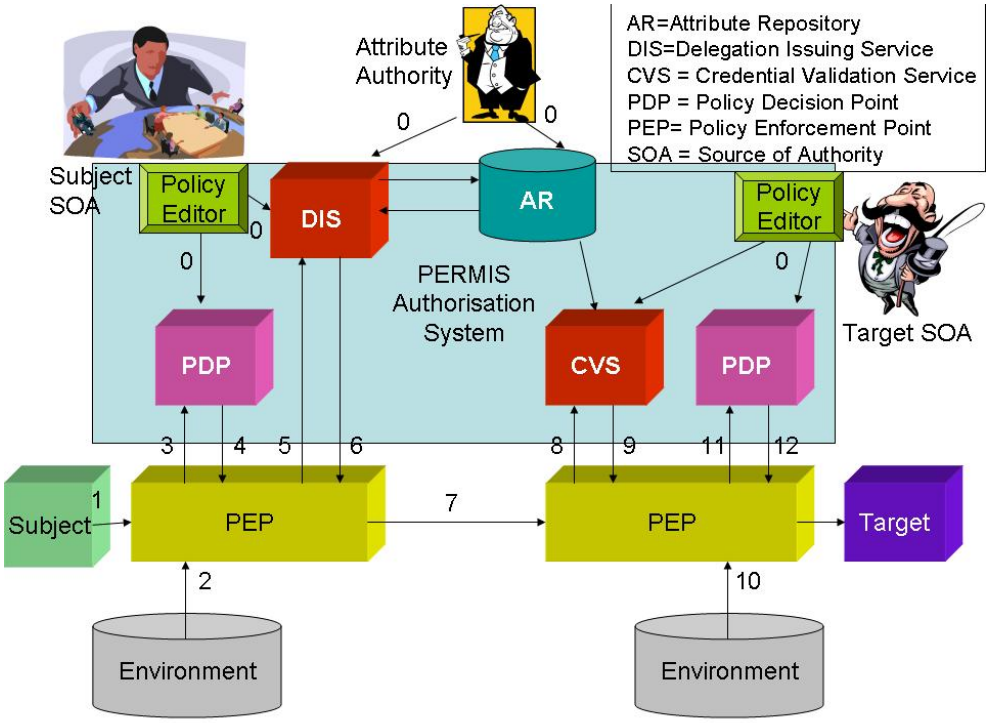

Figure 2. Components of the PERMIS Authorisation System

PERMIS implements the hierarchical RBAC model, which means that user roles (attributes) are organised hierarchically with superior roles inheriting the privileges of the subordinate ones. Each user is assigned one or more roles (attributes) as AC credentials, and each role or attribute is given a set of permissions in the authorisation policy that is written by the resource owner (the Target SOA in Figure 2). The PERMIS policy comprises two parts, a role assignment policy (RAP) that says who is trusted to assign which attributes to whom, and a target access policy (TAP) that says which attributes are needed to access which resources under what conditions (Chadwick, 2002). The credential validation service (CVS) of PERMIS evaluates all received credentials against the RAP,

discards ones that are not trusted, and passes all validated attributes to the policy enforcement point (PEP). The PEP, in turn, passes these to the PERMIS policy decision point (PDP), along with the user's access request, and any environmental parameters. The PDP makes an access control decision based on the TAP, and passes its *granted* or *denied* response back to the PEP. One can see that the PERMIS CVS is a PIP according to the Globus model, whilst the PERMIS PDP is a PDP. In order to gain access to a protected target resource a user has to present his credentials and the PERMIS decision engine (CVS and PDP) checks them against the policy in order to make an authorisation decision. The PERMIS toolkit provides an easy-to-use graphical user interface for creating its policies (the Policy Editor). Once created, the policies are converted into XML for input to the CVS and PDP. The policies may also be digitally signed by their authors and stored in attribute certificates, in order to stop them from being tampered with. They can also be stored in LDAP directories for subsequent retrieval by the PERMIS decision engine.

The PERMIS PMI architecture also consists of credential allocation systems, such as the Delegation Issuing Service, but these are not being used in this project and will not be described further. User authentication is not a function of PERMIS, and this is left to the particular application to enforce. As previously stated, the Globus Toolkit uses proxy certificates to authenticate the user who submits the Grid job.

PERMIS provides an API in the Java language for accessing the CVS and PDP, through calls to *getCreds* and *decision* respectively. The API caller provides the authenticated name of the user, who is identified by either his LDAP DN or his public key (proxy) certificate, to *getCreds.* The caller may optionally provide the user's credentials (the push model). Alternatively, the CVS can use the user's DN to retrieve the user's credentials stored in the configured LDAP directories (the pull model). After that the credentials are checked against the RAP and the valid roles and attributes are extracted and returned to the caller (the PEP). These are then passed to the PERMIS PDP via a call to *decision*, and PERMIS returns the "permit" or "deny" access control decision to the caller. This approach allows the system to control access to all resources in the target domain. The *getPBAAPI* function is used to initialise the PERMIS decision engine, and passes either the authorisation policy or the name of the trusted Target SOA and the URL where the policy can be found.

The use of PERMIS for authorising Grid jobs provides a flexible and efficient way of describing users' rights and permissions by means of its detailed authorisation policy. PERMIS can provide

both the PIP and PDP functionalities required by the Globus Toolkit. However, GT4 has recently integrated Shibboleth as a PIP, in the GridShib project, so these will be described next.

4. Shibboleth

Shibboleth is an Internet2/MACE project providing cross-domain single sign-on and attribute-based authorisation for systems that require inter-institutional sharing of web resources while preserving end-user privacy. Shibboleth software is already in use today and is being widely experimented with as an inter-realm access control solution for research and education applications. The main idea behind Shibboleth is that instead of having to log in and be authorised at each restricted site, users authenticate once at their home site, which then passes the user's attributes to the restricted sites in a privacy-preserving way.

The Shibboleth security protocol is based on SAML (Security Assertion Markup Language) assertions developed by OASIS. The Shibboleth specification (Erdos and Cantor, 2005) is a direct extension of the SAML 1.1 browser profiles (Maler et al., 2003). It includes two main software components: the Identity Provider (IdP) and the Service Provider (SP), which are deployed separately. The Shibboleth IdP creates and manages user identities and includes four primary components: the Attribute Authority (AA) which releases attributes to the SP Attribute Requester; the Handle Service (HS) which generates random temporary handles to be used in communication with the SP thus preserving the user's confidentiality; the directory service which stores user attributes; and the local single sign-on (SSO) system. The Shibboleth SP controls access to its resources by requesting and consuming the necessary attributes from the IdP.

When the Shibboleth SP receives a request from the user for a Shibboleth-protected resource, it redirects him to a "Where are you from?" service, which asks the user to select the organisation he is from. After that the user is redirected to this organisation's authentication system via the site's Handle Service, which interacts with the local authentication system to authenticate the user, and then generates a random handle to return to the SP. The SP can use this handle to request the user's attributes from the IdP and according to these attributes, it can decide whether to grant or deny access to the resource. The IdP AA can use the existing institutional identity management infrastructure, such as an LDAP directory, for storing the attributes. A Shibboleth authentication and authorisation session consists of a cookie, which is associated with a security context, being passed between the user's browser and the various web servers.

Shibboleth provides an efficient means of identity federation and inter-institutional authorisation with an emphasis on user privacy protection and management. It can be used to add attribute-based authorisation to the Globus Toolkit. The GridShib project described next is being developed to allow the Shibboleth IdP service to provide the user's attributes, which will then be used to authorise Grid jobs in a VO context.

5. GridShib

GridShib is a research project, currently being undertaken at the University of Illinois, the University of Chicago and Argonne National Laboratory that allows for interoperability between the Globus Toolkit and Shibboleth (Barton et al., 2006). The first release of the GridShib software consists of separate plug-ins for the Globus Toolkit and the Shibboleth IdP and was made available in September 2005. It provides the Globus Toolkit Grid Service Provider with the means to securely request a user's attributes from the Shibboleth IdP. They are collected by a GridShib Policy Information Point plugin that is capable of communicating via the Shibboleth protocol to the Shibboleth IdP and fetching the SAML attribute assertions. These are then parsed, the attributes extracted and passed back to the GT4 PEP. The latter relays them to a configured Policy Decision Point (PDP) which makes an authorisation decision based on the Shibboleth attributes. This separation of PIP and PDP allows for the flexible use of plug-ins and is used in our project in order to combine the Shibboleth attribute collection carried out by GridShib and the authorisation decision-making carried out by PERMIS.

A large challenge in integrating Globus Toolkit with Shibboleth is that different IdPs use different name forms and name values to identify the same user. Users are identified in GT4 by X.500 distinguished names (DNs), and in Shibboleth by opaque handles. The GridShib team overcame this incompatibility issue by introducing a DN mapping plug-in for the Shibboleth IdP, which maps DNs into Shibboleth handles. Unfortunately this has significantly constrained the scalability and manageability of the GridShib software and so this is a major issue that is currently being resolved.

The focus of the GridShib project was to leverage the attribute management infrastructure of Shibboleth, by transporting Shibboleth attributes as SAML attribute assertions from the Shibboleth IdP to any Globus Toolkit-based decision making service (PDP) (Welch et al., 2005b). Consequently, it provides limited PDP functionality itself. The GridShib PDP makes the

authorisation decision by simply parsing the attributes and comparing them to an access control list that contains the allowed attributes. Each attribute is then mapped into a local account, which is analogous to the current *grid-mapfile* functionality. This leads to users being granted access if they have any single attribute in the access control list. Consequently the GridShib PDP is not capable of making decisions based on combinations of attributes (e.g., member of University X and project Y), or on dynamically changing conditions, such as the time of day, the amount of resources being consumed, or even on parameters of the user's request, such as the operation, the requested target or the job priority. Furthermore, it is not able to check if the IdP issued the correct set of attributes that it is trusted to issue (although this is a feature that the Globus team plan to add).. The PERMIS decision engine, on the other hand, is capable of making such complex decisions, since its PDP and CVS are policy driven and can test for arbitrary conditions being fulfilled before access is granted. We will now describe our project, which was based on the first release of the GridShib project and was carried out in collaboration with the GridShib team.

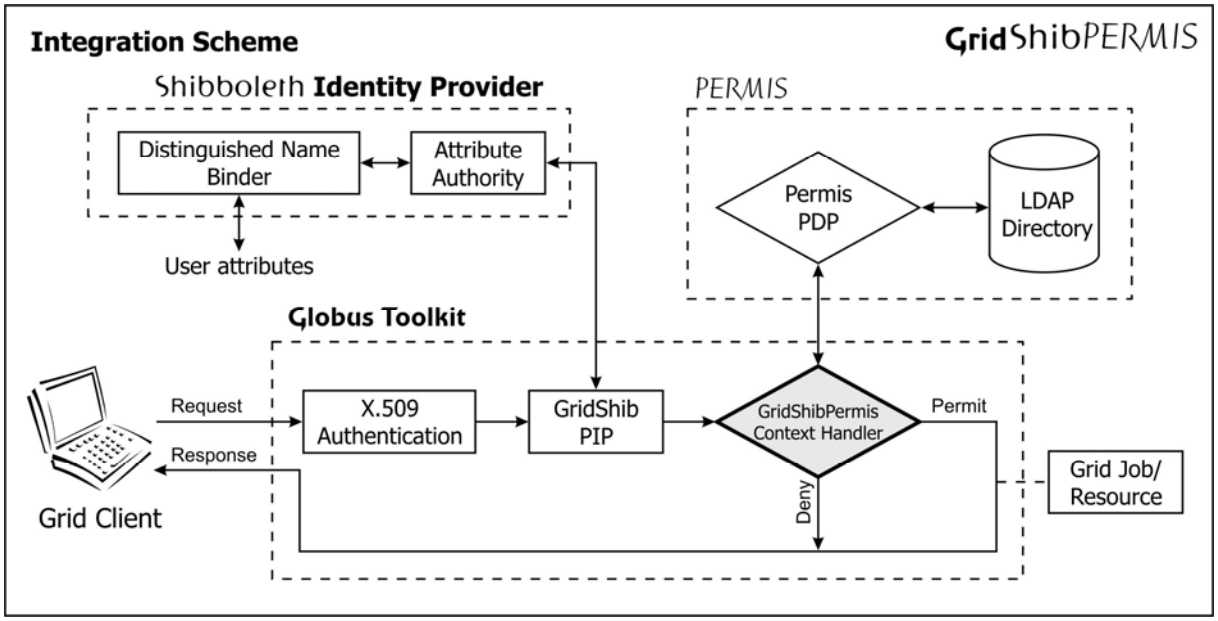

Figure 3: GridShibPERMIS Integration Scheme

6. GridShibPERMIS

In order to carry out the GridShib and PERMIS integration, the GridShibPERMIS Context Handler was written to interface to PERMIS, and to appear as a PDP in the Globus Toolkit authorisation framework. As already mentioned, the Globus Toolkit GSI authorisation framework allows custom PIPs and PDPs to be included in the authorisation chain through callouts to external authorisation services. This chain, with relevant configuration details, can be configured at resource, service or

container level. The authorisation chain is invoked immediately after the authentication of the request is finished and evaluates each PIP and PDP in the order it is specified in the chain. The decision is made on the basis of a deny-override mechanism (Freeman and Ananthakrishnan, 2005), which means that if any PDP returns a *deny,* the chain processing immediately stops. Globus Toolkit GSI demands that PDPs must implement the interface *org.globus.wsrf.security.authorization.PDP* and return a permit or deny decision on the basis of the subject's DN, message context and the requested operation. PIPs, in turn, have to implement the *org.globus.wsrf.security.authorization.PIP* interface, and must place the set of retrieved attributes in a pre-determined place where the PDPs can find them. The GridShib PIP is used as the latter, and the GridShibPERMIS Context handler as the former. Since PERMIS already contains its own PIP (the CVS) as well as a PDP, one might well ask what purpose the PERMIS CVS now serves? This will be explained next.

A large challenge with the Globus Toolkit, Shibboleth and PERMIS integration is that there are currently no unified standards for the format of the attributes, or for the credentials which securely embed the attributes for transfer from the attribute authority to the relying services. Both X.509 ACs and SAML attribute assertions are alternative encodings for user credentials. They both essentially contain the name of the user, the set of attributes, the name of the issuer asserting that this user has this set of attributes, a time validity period and the digital signature of the issuer. In PERMIS, X.509 ACs are passed to the PIP (CVS), whilst in GridShib, Shibboleth SAML assertions are passed to the PIP. Whilst X.509 ACs are a binary encoding of an ASN.1 data-type, SAML attribute assertions are a string-encoding of an XML data-type. Of course, there are some other differences in the finer details of each, but conceptually they serve the same purpose. One of these differences is that the digital signature is mandatory in ACs and optional in SAML assertions. Shibboleth chooses to omit the signature, and instead the SAML assertions are protected by an SSL connection between the IdP and the SP. This effectively means that the SSL key pair owner is a proxy signer, signing all the SAML attribute assertions on behalf of that site. This effectively precludes the distributed management of attributes and the delegation of authority, since one authority is signing for all. In the PERMIS model, there can be multiple AAs per IdP, and each signs the ACs that it issues. Consequently this supports both distributed management and dynamic delegation of authority.

Shibboleth attempts to provide some support for the distributed management of attributes, by allowing the origin site to append a *scope domain* to each released attribute value. Scope domains are used to distinguish between different attribute issuers at the origin site, for example some

attributes could have a scope domain of "@kent.ac.uk", while others could have a scope domain of "@computing.kent.ac.uk" (note that the same attribute cannot have multiple scope domains, which effectively precludes dynamic delegation of authority). When the PERMIS CVS is passed "scoped" attributes instead of digitally signed ACs, the scope domains take the place of the AC signers (i.e., Subject SOAs in Figure 1). In order to validate "scoped" attributes, the PERMIS RAP specifies the scope domains as SOAs in place of AC issuer DNs. We have reserved a special URL "shib:<scope domain name>" for this. In the above example, there would be two corresponding SOAs identified in the RAP by the special URLs: "shib:kent.ac.uk", and "shib:computing.kent.ac.uk". The scoped attributes can now be validated against the RAP by the PERMIS CVS in the same way as X.509 AC issuers, except that cryptographic validation cannot be performed. Thus, there is no proof of who actually issued the attributes, because "scoped" attributes do not have digital signatures. If scope domains are not being used by an origin site, then the GridShib PIP inserts the name of the origin site as the scope domain for all the attributes, but only if its name is configured into the PIP's SAML2 metadata. This solution is very constraining since it only allows for a single origin site to be configured.

Concerning the format of the attribute values, there is no single standard for this, although LDAP is the most prevalent. Both Shibboleth and PERMIS use attributes stored in LDAP format in directories. Shibboleth converts these into SAML format for transfer, whilst PERMIS embeds them in X.509 ACs. The GridShib PIP, on the other hand, passes attribute information to the PDP by way of a plain Java object, and so the GridShibPERMIS Context Handler was built to support this.

Concerning the content of attribute values, this turns out to be a non-issue, since PERMIS policies can accept any content as long as it can be represented as a string, and Shibboleth transfers all its attribute values as strings. Various LDAP RFCs describe how to format attribute values of different syntaxes into strings (e.g., Wahl, 1997; Hodges, 2002)

The Globus authorisation chain is configured to invoke the GridShibPERMIS Context Handler after the GridShib PIP. The Context Handler extracts the attributes returned by the GridShib PIP, and converts them into the internal Java format recognised by PERMIS. This was done by extracting and parsing the name-value pairs. The Context Handler then passes these attributes to the PERMIS CVS which checks if the IdP is trusted to issue the attributes it has returned, according to the PERMIS policy. Next the valid attributes are passed along with information about the user's requested action and target to the PERMIS PDP via a call to *decision*. The PDP makes an access control decision according to the configured PERMIS policy and this is returned by the Context

Handler to the GT4 PEP. The GridShibPERMIS Context Handler also allows X.509 ACs to be returned by Shibboleth, for those sites that support them, and these are passed to the PERMIS decision engine unchanged. Figure 3 summarises the integration scheme.

7. Conclusions and Future Work

The prototype GridShibPERMIS software has been developed and released for download at the PERMIS web site (http://sec.cs.kent.ac.uk/permis). It requires the Globus Toolkit v4 with both GridShib and PERMIS installed and provides the GridShibPERMIS Context Handler for access control decision making. This has to be configured to be called after the GridShib PIP.

The GridShibPERMIS project provides an integrated role-based access control solution for the Globus Toolkit. It provides the advantages of both Shibboleth cross-organisation identity federation and attribute management with PERMIS policy driven role-based access controls, giving the necessary functionality, flexibility and fine-grained access control for authorising Grid jobs. However, there are still a number of issues to be resolved in the forthcoming and future releases of the software.

The current method of attribute communication between Shibboleth, Globus Toolkit and PERMIS does not yet support one of the main Shibboleth advantages – pseudonymous access to Grid resources, as the user's DN and attributes are currently released to the Grid application during the process of authorisation. One of the methods proposed by the GridShib team to solve this problem is to include the GridLogon service in the authorisation framework. This would act much like the Shibboleth Handle Service providing the authenticated user with a set of credentials with a pseudonym identifier.

As mentioned earlier, there is currently a scalability problem because the Shibboleth IdP plugin requires each Shibboleth handle to be mapped into the equivalent Globus DN. The Globus team are working on a solution which will use an online CA and newly-minted certificates.

There is also a limitation that prevents users with attributes assigned by different AAs from merging them and using them to gain access to a resource. We have a proposal for resolving this which involves the user linking his attributes together in the different AAs that hold them, via randomly generated identifiers. When the SP contacts the AA/IdP to request the user's attributes, not only are

the locally held attributes returned, but the AA also returns referrals which point to other AAs that hold linked attributes of the same user. These referrals are encrypted for the linked AAs, so that the SP cannot see inside their contents, and cannot therefore link together the user's various identities at the different AAs (Chadwick, 2006).

Another issue is validation of the attribute authorities of a user in the Grid context. As described above, this can be done if the scope domain is included in the attribute values by the IdP, but if it is not, this requires the GridShib PIP to either add it to the attribute values or pass it to the Context Handler so that the PERMIS CVS can validate it. The current limitation is that only one issuer scope domain is supported through the PIP configuration file. It is proposed to make this more dynamic and infinitely extensible by adding the IdP *providerId* to the authorisation request and attribute issuer property to the SAML assertion response. This will be carried out in collaboration with the GridShib team.

In summary, the GridShibPERMIS project implements fine-grained policy-driven role-based access controls for use in the authorisation of Grid applications. We believe that the results of this project will provide a robust and integrated security infrastructure for both Grids and campus applications, having low deployment costs, because it leverages the existing Shibboleth and PERMIS infrastructures to provide policy-driven role-based privilege management for both Grids and campus-based applications.

Acknowledgements

The Authors would like to thank the UK JISC for funding the GridShibPERMIS project. We also thank Tim Freeman and the GridShib development team for their continued cooperation and support.

References

Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, Á., Gianoli, A., Lörentey, K., and Spataro F. (2003) "VOMS: an Authorization System for Virtual Organizations", 1st European Across Grids Conference, Santiago de Compostela.

Barton, T., Basney, J., Freeman, T., Scavo, T., Siebenlist, F., Welch, V., Ananthakrishnan, R., Baker, B., and Keahey, K. (2006) "Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, Gridshib, and MyProxy", 5th Annual PKI R&D Workshop (To appear).

Chadwick, D.W. (2001) "An X.509 role based privilege management infrastructure", Business Briefing - Global InfoSecurity 2002, World Markets Research Centre Ltd.

Chadwick, D.W., and Otenko, A. (2002) "RBAC Policies in XML for X.509-based Privilege Management", Proceedings of IFIP 17th International Conference on Information Security (SEC 2002), Kluwer Academic, pp. 39-53.

Chadwick, D.W., and Otenko, A. (2003) "The PERMIS X.509 role based privilege management infrastructure", Future Generation Computer Systems, Vol 19 No 2, pp. 277-289.

Chadwick, D.W. (2006). "Authorisation using Attributes from Multiple Authorities" in Proc 15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WET ICE 2006), 26-28 June 2006, Manchester, UK.

Erdos, M., and Cantor, S. (2005) "Shibboleth-Architecture DRAFT v05", Available http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-v05.pdf.

Foster, I., Kesselman, C., and Tuecke, S. (2001) "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of High Performance Computing Applications, Vol 15 No 3, pp. 200–222.

Foster, I., and Kesselman, C. (Eds.) (2004) "The Grid 2: Blueprint for a New Computing Infrastructure", Morgan Kaufmann.

Foster, I. (2005) “Globus Toolkit Version 4: Software for Service-Oriented Systems”, IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp. 2-13.

Freeman, T., Ananthakrishnan, R. (2005) “Authorization Processing for Globus Toolkit Java Web Services”, IBM Developer Works, Available http://www-128.ibm.com/developerworks/grid/library/gr-gt4auth/.

Hodges, J., Morgan, R. “Lightweight Directory Access Protocol (v3): Technical Specification”, RFC 3377, September 2002.

ISO 9594-8/ITU-T Rec. X.509 (2001) The Directory: Public-key and attribute certificate frameworks

Maler, E., Mishra, P., Philpott, R. (2003) “Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML) v1.1”, OASIS, Available http://www.oasis-open.org/committees/download.php/3405/oasis-sstc-saml-bindings-1.1.pdf.

Morgan, R.L., Cantor, S., Carmody, S., Hoehn, W., and Klingenstein, K. (2004) “Federated Security: The Shibboleth Approach”, EDUCAUSE Quarterly Vol 27 No 4, pp. 4-6.

Security Association Markup Language (SAML) Specification (2005), Available http://www.oasis-open.org/committees/security/.

Thompson, M., Johnston, W., Mudumbai, S., Hoo, G., Jackson, K., and Essiari, A. (1999) “Certificate-based Access Control for Widely Distributed Resources”, 8th Usenix Security Symposium.

Tuecke, S., Welch, V., Engert, D,. Pearlman, L., Thompson, M.. “Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile”, RFC3820, June 2004.

Wahl, M., Coulbeck, A., Howes, T., Kille, S. “Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions”, RFC 2252. December 1997

Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L., and Tuecke, S. (2003) “Security for Grid Services”, 12th IEEE International Symposium on High Performance Distributed Computing.

Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S., and Siebenlist, F. (2004a) “X.509 proxy certificates for dynamic delegation”, Proceedings of 3rd Annual PKI R&D
Workshop.

Welch, V., Siebenlist, F., Chadwick, D., Meder, S., and Pearlman, L. (2004b) “Use of SAML for OGSA Authorization”, Global Grid Forum.

Welch, V. (Ed.) (2005a) “Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective”, Available http://www.globus.org/toolkit/docs/4.0/security/GT4-GSI-Overview.pdf.

Welch, V., Barton, T., Keahey, K., Siebenlist, F. (2005b) “Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Collaboration”, Proceedings of 4th Annual PKI R&D Workshop.

Brief autobiographical note

David W Chadwick is Professor of Information Systems Security at the University of Kent, and leader of the Information Systems Security Research Group. His primary research interests are: public key infrastructures, privilege management infrastructures, policy based security, trust management, and autonomic security. His full address is The Computing Laboratory, University of Kent, Canterbury, CT2 7NF. Tel: +44 1227 82 3221 Mobile: +44 77 96 44 7184
Email: D.W.Chadwick@kent.ac.uk

Andrey Novikov is a Diploma in Computer Science student at the University of Kent. He is also a final year Computer Science student at the Samara State Technical University, Russia. E-mail: an64@kent.ac.uk. Full international address: B3-7 Keynes College, University of Kent, Canterbury, Kent, CT2 7NP.

Alexander (Sassa) Otenko is a research fellow at the University of Kent. He has PhD from the University of Salford and a specialist degree in Applied Mathematics and Computing from Sumy State University in the Ukraine, where he was the top student in his year. He was the main architect and Java programmer on the PERMIS project. His full address is: The Computing Laboratory, University of Kent, Canterbury, CT2 7NF. Email: o.otenko@kent.ac.uk