Federated Grids and their Security

Geoffrey Fox and Marlon Pierce

Draft 0.4

Introduction

We examine issues involved in creating "virtual Grids" out of specific Grid installations. We see it as inevitable that Grids will be established as islands of resource collections that will need to be federated in lightweight manners. These islands of resources may result from a number of reasons. First, a group may set up a Grid of resources that it owns and directly controls but must overcome sociological and /or organizational barriers for interoperating with other Grid installations, even when the same software is used. Second, we see the emergence of Grids built out of differing substrate technologies with perhaps incompatible communication systems.

As a specific example of the first case, we think of a Grid as a single installation of Globus software on some specific resources. For example, the Community Grids Lab may install Globus software on all of its Linux and Solaris servers. The CGL sets up a Grid consisting of resources that it owns and has direct control over. We then must negotiate with other Grid operators (such as in the Computer Science department) about federating these Grids. Currently this is an all-or-nothing affair, but in this white paper we examine some possibilities for adding more fine-grained control.

As an example of the second case, the Open Grid Services Architecture defines Grid services and provides a roadmap for building sophisticated, interacting Web Services by providing extensions to the Web Service Definition Language. One of the important (and currently under development) extensions is the notification framework that allows Grid services to communicate state changes with each other. Even though there will be standard interfaces for these notification services, the candidate existing implementations (such as Java Messaging Service and IBM's MQ Series) are not compatible. It will therefore be necessary to federate Grids with different notification and messaging implementations.

The purpose of this report is to examine the consequences, requirements, and possible implementation issues needed to support security in federated grids, with a primary focus on building overlay networks to simply federation and limit access and security liabilities. Before proceeding, we first note that there are other aspects of federated Grids that are described elsewhere. Principally, Virtual Organizations require federated roles in such systems provide the scalability and loose coupling needed for federation. This is discussed in detail by Chivers [1]. We note here that user identity in current Grid security implementations has two major shortcomings: it does not scale and it undermines traditional Web security by effectively creating a super-Virtual Private Network that includes users outside the direct control of the constituent (real) member organizations of a VO.

This report focuses on the Grid topologies that must be present to support security in federation. We first note the resemblance of standard Grid security to Virtual Private Networks, but as we point out this actually subverts know best practices for commercial Web security. We instead propose a hierarchical federation of grids controls through various software routers, creating an overlay network that may be used to control the flow of messaging in the system between different grids. Such arrangements are known to be efficient for routing messages, but they also allow us to define hierarchical rings of security that limit the liability of compromised authentication.

We are particularly interested in using authorization to mitigate the risk for trusting authentication from users outside of the control of a particular real organization. We focus particularly on what we believe is the middle ground between the realistic secure deployment scenarios (with lessons learned from the e-commerce world) discussed in [2] and the finer-grained policy issues addressed by such systems as Akenti and CAS. We particularly are interested in using a distributed publish/subscribe metaphor to enable simple but powerful distributed authorization systems for federating Grids. Such a system may be used to create a "layered virtual network," in which we are able to replicate in application software some of the best practices from the commercial sector (VPNs, Firewalls, DMZs). The express purpose of such as system is to develop rings of security both within Grids and around Grids in a lightweight fashion, without requiring, for example Firewall and/or network router modifications for every VO membership or policy change.

Before proceeding, we note that throughout this report we refer to "a Grid" and "Grids" to mean specific installations of Grid software onto resources owned by a single, real organization. In contrast, "the Grid" as a single, globally distributed computing infrastructure is considered to be composed of a patchwork of Grid installations. We note that the emergence of a single Grid might be comparable to the emergence of the Internet, in which routing/mediating entities negotiate the transversal of Grid messages across heterogeneous boundaries, providing a veneer of uniformity through simple encapsulating and translating mechanisms.

Grid Authentication and Authorization

The Grid Security Infrastructure (GSI) defines the standard for Grid authentication. GSI is technically mechanism independent, building around the GSSAPI. However, the standard implementation is based around PKI, and while it is possible to set up pure Kerberos Grids, this typically requires a great deal of effort.

GSI is essentially a single-sign on mechanism that allows users to acquire login credentials from one Grid machine and via delegation access all machines on the Grid. In this, GSI strongly resembles Kerberos, with the notable exception that Kerberos defines realms of security. Typically a Kerberos realm would be a department or campus-wide computer network. Kerberos realms can be configured to trust each other (cross-realm authentication) in a scalable way (as of Kerberos v5). The Department of Defense High Performance Computing Program is an example of one large deployment, with four realms and a few thousand users. Both security mechanisms enable (via ticket/credential forwarding) a user (or application) to transverse several different sites. This is typically useful when computers in realm have different installed software, don't share file systems, have different capabilities, etc.

GSI differs from Kerberos in intent in that it is designed (through PKI certificate authorities) to more simply enable inter-organizational authentication—Kerberos cross-realm configurations are rather static.

Basic Grid security thus relies heavily on authentication, with implicit trust in underlying authorization models of the Grid-enabled resource. Essentially, a remote user's credentials are mapped to an account on a particular machine. The remote user then has all of the permissions of that account. This was often euphemistically described as allowing the resource sites to set their own usage policies: each site governs the rights it grants to a particular user after authentication.

Several more sophisticated authorization services (such as Akenti from LBL and more lately CAS form the Globus team) have been developed. These provide policy languages and service implementations that provide an additional layer of security: just because we trust the credentials of a user does not mean we trust that user with everything. Akenti focuses on access control to specific resources while CAS is concerned with access control to groups of resources. Both systems rely on X.509 certificates to convey policy assertions and attributes. CAS in particular is "policy-language neutral" and is designed only to carry policy statements, which must be interpreted by an appropriately implemented CAS server. Such authorization systems also have the advantage of enabling a more coherent usage policy for a VO. That is, without policy services, one partner in a VO may set up extremely restricted accounts for Grid users, while another partner may give its Grid users all normal user privileges. To avoid this undesirable situation, policy would have to be negotiated between partners "off-line" in an ad-hoc fashion.

Web service solutions for conveying security policies will inevitably have an impact on Grid services as current Grids evolve into OGSI implementations. Web services communicating with SOAP have an open-ended ability to encapsulate other XML messages through one or more SOAP headers and the SOAP body, so the main XML payload (in the body) may be supplemented with several different XML messages describing authentication and authorization assertions. WS-Security, WS-Policy, and SAML are all examples of these. It remains to be seen how these commercial standards will be integrated into Grid implementations.

Relying on Grid authentication alone has several dangers. First, as with other single sign on mechanisms like Kerberos ticket forwarding, users or applications create a string of proxy credentials on machines as they move about the system. These proxy certificates have a limited lifespan of a few hours, but if stolen allow the thief to impersonate the user during that time. More seriously, a user's account on PKI-based Grids is only as secure as the user's permanent (or long-term) private key. PKI systems rely on revocation lists to identify compromised permanent keys, but RLs are notoriously non-scalable in practice.

Virtual Private Networks and Grids

Broadly speaking, a VPN is a network tunneling mechanism that connects distributed resources securely into a private network by using encryption technologies rather than actual physical network lines. In practice we may think of these as virtual private intranets—the VPN provides access to restricted networks to computers outside the restricted intranet. Secure connections are made at the message level rather than through secure transport links, so the message can securely transverse an untrusted network. Trust here is implicitly all or nothing. The source and destination have decided to trust each other completely. Nothing else is trusted, including the VPN service provider. As we shall see, VPNs are very similar to "single-hop" Grids.

VPNs may be implemented in a number of ways, but for concreteness we consider first a network-level implementation. We may set up a VPN between two specific routers. Incoming IP messages for the VPN are encapsulated inside another IP message that is sent between the starting router and ending router. When the wrapped message is received by the endpoint router, the original message is unwrapped and the router sends the original IP message on to its destination.

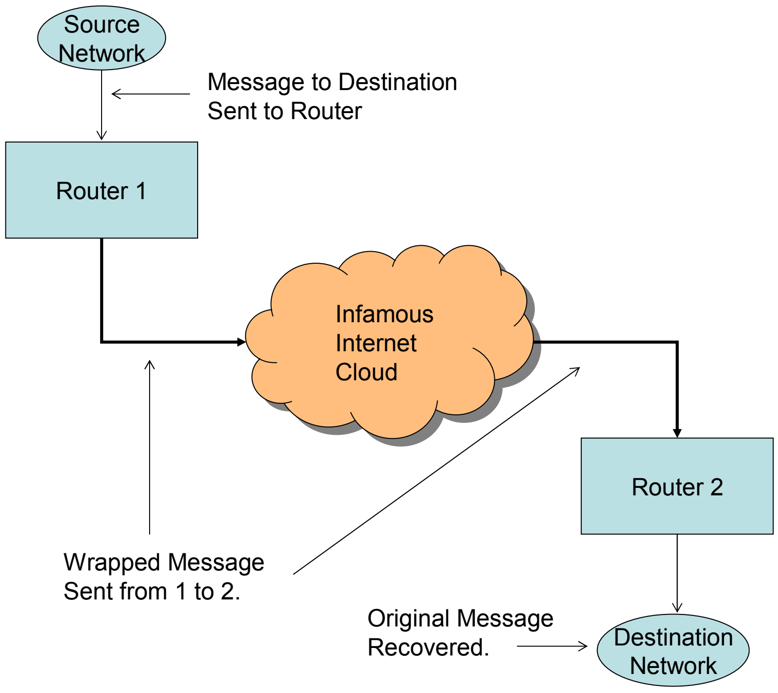
Source Network
Message to Destination Sent to Router
Router 1
Infamous Internet Cloud
Router 2
Wrapped Message Sent from 1 to 2.
Original Message Recovered.
Destination Network

Figure 1

Here, Router 1 receives messages from the Source with an endpoint in the Destination Network. The wrapped message in the intranet cloud is set as an IP message between routers.

Such routing has at least three possible (and possibly overlapping) uses:

1. When combined with message-level security mechanisms (encryption, digesting, signing), it can provide a secure private link, assuming some initial step of setting up private keys has taken place. The security mechanisms may be implemented in either hardware or software. IPSec, for example, may provide the security mechanisms.
2. It can be used to provide direct connections between networks with enhanced capabilities (such as multicast).
3. It can be used to wrap non-IP messages, assuming the destination knows how to handle the custom protocol messages after they are unwrapped. As we shall see, this metaphor has an important extension in Grid federation.

As described, this approach assumes a network-level routing configuration. That is, the network administrators actually modify their routing tables to create the virtual network interface. However, the general ideas should be also applicable to Grid software application level routing.

In passing, we should clarify our point of view about Grid networking issues: while security at the network level should be incorporated into federated Grids, we are more interested in application software that can create "virtual routers" over the top of the real network. Such systems can be configured by Grid administrators, decoupling the mechanics of Grid administration from hardware administration and network administration (although such groups obviously must cooperate and will typically have overlapping members).

Applications and Limitations of VPNs for Grids

We may easily change Figure 1 so that the Source and Destination Networks are instead Source and Destination Grids, and the Routers are renamed to Grid Routers (GRs)—that is, application software-level routing applications. One may then think of Figure 1 as constituting a GridVPN, which may be built on top of other network-level security mechanisms. We may think of the GRs as specialized peer brokers that route messages between the source and destination Grids. The separate Grids may be set up to ignore all outside traffic, so that the GR acts as a proxy server. That is, if I am outside of the Destination Grid, I may not talk directly to the Globus Gatekeeper on port 2119 of some server. I instead have to talk to the GR on some other port on some other host, which routes my request if all goes well. This would allow Grid administrators to maintain "standard operating Grids" of their own resources and users, but by configuring the GR, they may quickly turn on (and turn off) access to their Grid resources from other Grids.

As described, VPNs incorporate authentication and secure transmission. This has three problems. First, we must trust the authentication mechanism. Users (typically by possessing access to the right private key) can prove their identity to the VPN and are thus allowed to send messages. The VPN then makes sure the message goes to the right place securely. This is fine as long as the organization that maintains the Grid has strict control over the private keys as well. However, Grids imply collaboration between separate organizations, thus in affect requiring trust in the maintenance of private key security in users from other organizations. Thus, the destination Grid in Figure 1 must trust that all the users of the Source Grid have properly secured their private keys.

Second, the VPN itself does not make any access control decisions. Such decisions are usually made by firewalls that are coupled to the routers. Firewalls are discussed below, although the line between the VPN and firewalls for Grids is blurred.

Third and finally, there is one single, homogenous security realm, when in practice we know that we must instead have rings of security that can quickly seal off and isolate compromised areas. To address this issue, Grids actually will need to be built up out of several hierarchical VPNs. We examine this issue in the next section.

Hierarchical VPNs

In the standard VPN description, the Source and Destination networks of Figure 1 are physically private networks, so only one VPN is needed. For Grids, however, this may not be the case. Broadly, a Virtual Organization may itself be composed of smaller Virtual Organizations. Thus in the Grid analogy to Figure 1, the Source and Destination may themselves be federated networks. Also, of course, we may wish (even within a Grid of our own resources) to subdivide resources into smaller subunits for the purposes of access control and "firewall" security. That is, in our Grid way may have some resources that are for internal use only (private), some that are for trusted partners (shared), and some that are more openly accessible from other Grids in a test bed.

Thus, our single Grid installation (or a VO that is part of a larger VO) must enforce multiple levels of access and protect different access partitions from compromises in the other partitions. For example, the testbed may include some resources that we own that we are not overly concerned with. We want to prevent them from being hacked, but if they do get hacked it is not a disaster. Such test beds would need to be thoroughly "sandboxed" and we should be able to quickly pull the plug on them if they are compromised without bringing down the rest of our Grid.

We may represent this situation by magnifying one of the Grid bubbles in Figure 1, as shown below.

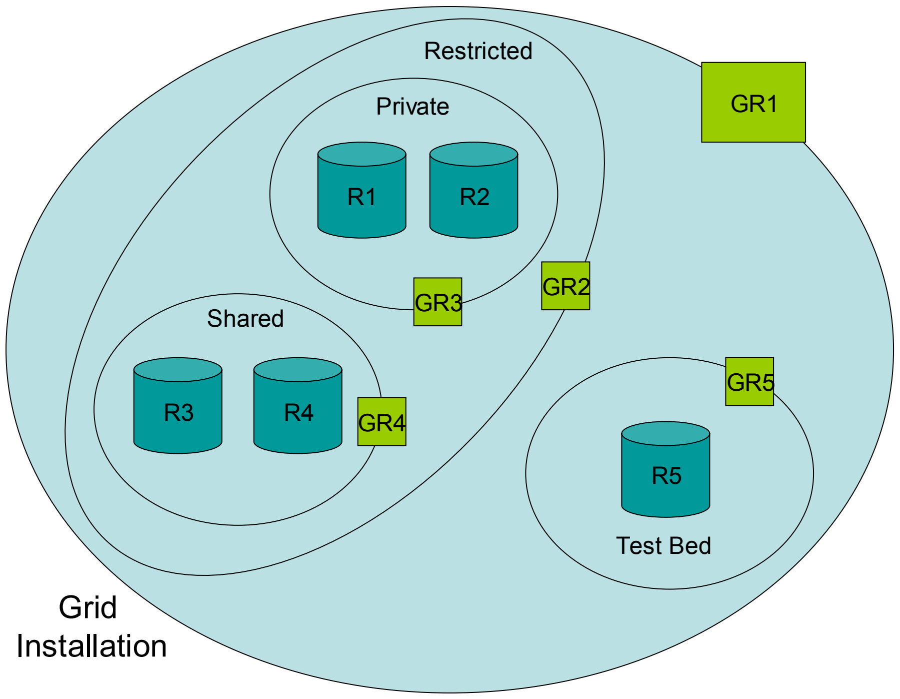

Figure 2

The layered Grid described above may be configured as in Figure 2. The Grid installation itself is accessed at the perimeter through a Grid Router, GR1, which forwards all incoming messages to the internal Grid Routers. Messages without proper destinations and credentials are rejected here. Within the Grid itself, we have Resources R1-R5 that are partitioned as shown. Access to these resources is controlled by one or more internal Grid Routers. For example, the Shared and Restricted resources are separated from each other by GR3 and GR4 and both separated by an additional security wall, GR2, from the Test Bed. Thus if the Test Bed's security were compromised, it can be isolated from both the internal and external resources by shutting down its Grid Router, GR5. Such changes need to be propagated to the other Grid Routers.

This figure is drawn with NaradaBrokering in mind. Here, incoming requests for resources are divided logically into topics and "physically" into different brokers running on different machines. A single publish/subscribe realm is distributed across several brokers for load balancing and fault tolerance, as well as security. Incoming requests are matched to publication privileges. For example, I may have publication privileges to the topics "Test Bed" and to "Restricted/Shared" but not to the topic "Restricted/Private", so I can assess Resources R3-R5 within the Grid for executing services.

In Figure 2, the space between the "restricted" area and the open area that includes the Test Bed is often termed the *demilitarized zone (DMZ)*. The DMZ separates the open internet from private intranets and often hosts web servers. Again, here we mean effectively a "Grid DMZ" that is defined by our Grid Routers. This may be built over the top of other network-level DMZs.

In any case, it is assumed that the Grid Router uses some authentication mechanism. If this is PKI based, for example, the system is still susceptible to the vulnerabilities of compromised keys, as described above. The difference now is that one has some control over the amount of trust placed in different users by strictly limiting resources they can access.

Firewalls

The routers shown in Figure 1 for typical VPNs are often coupled with firewalls, which may serve a couple of purposes. First, firewalls may act as proxies and gatekeepers, single entry points for restricted resources. Thus protected resources do not need to be accessed directly from the outside and are instead only accessed by a trusted gatekeeper. This provides the very practical advantage that in cases of security emergencies, the restricted resources can be protected immediately by shutting down the gatekeeper. For endemic security problems, the gatekeeper can remain up while denying access to known untrusted sources or perhaps denying access to all except a few privileged outside sources.

As shown in Figure 2, firewall protection can be inverted, surrounding and cutting off access to compromised resources, preventing the compromised resources from attacking other resources.

Firewalls may also provide an additional level of security by filtering. For Grid applications, we may need to provide sophisticated protocol filtering that examines the actual content of protocol to make sure that it is a valid message. For example, in Figure 2, an encrypted message may be destined for private Grid through a “GridVPN” style setup. The Grid Routers do not examine the content of the message itself, but simply forward it until it reaches GR3 that guards the private resources. This GR can then actually inspect the content of the message and verify that message is in fact a valid application protocol (SOAP, for example).

Demilitarized Zones (DMZs)

One of the commercial world’s established best practices that should be adopted by production Grid systems (particularly federated Grids with limited trust) is the establishment of DMZs that separate the open Internet from the corporate intranet. DMZs serve as a buffer region where publicly accessible resources may be placed and reasonably controlled, but which do not (if compromised) compromise the security of internal, private resources. DMZs are typically built with firewalls. A VPN is essentially the part of the private corporate network. The DMZ is typically the location of Web servers and related databases, and as discussed in [2] is also an appropriate place for many Grid servers. A typical DMZ is shown in the figure below:

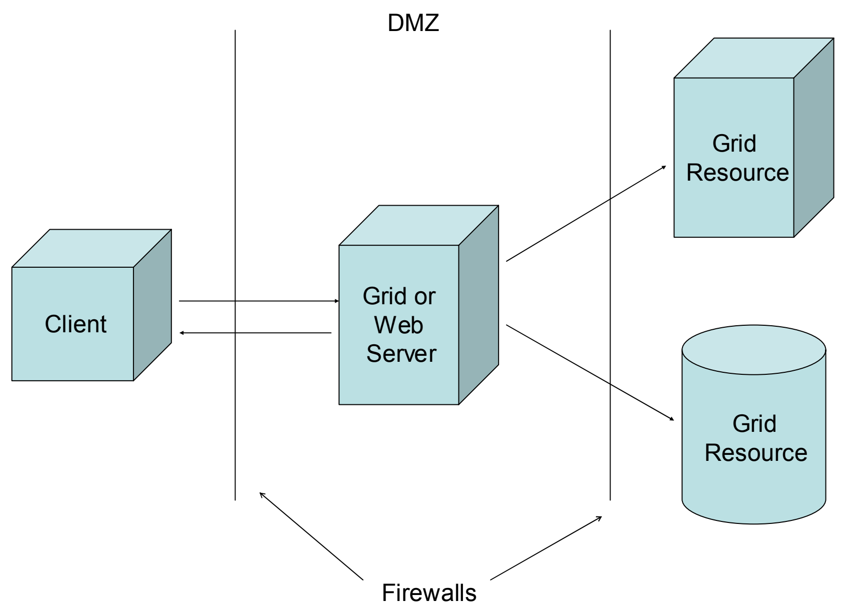

Figure 3

The basic idea is that the client must cross a protective firewall boundary in order to reach the Grid service running in the DMZ. This protects both the client and the server: if either is compromised, the firewall can temporarily block access until the problem is resolved. Note in the figure that the private intranet is located on the right-hand side. It may consist of resources on both real and virtual private networks.

The problem with Grid security technologies is that they effectively act like super-VPNs that can potentially turn the entire Grid into a single intranet. Instead, the security set up should resemble that shown in Figure 3. Here the Grid clients primarily interact with a Grid resource located in the DMZ. This Grid resource (or collection of resources) may provide all the interactions that are needed by the client. This resource may however need to interact with Grid resources inside the private network (either by push or pull mechanisms) on behalf of the client. By imposing the intervening firewall, we enable the client to access the private resources indirectly in a strictly controlled fashion, and we can (in emergencies) cut off that access. More sophisticated firewalls may also impose more selective control (and denial) of access to particular clients.

The above scenario is a useful "best practice" that should probably be followed by groups deploying production Grids. Here the DMZ is the organization's preexisting DMZ, and the firewalls are standard commercial firewall products. However, we believe that there is also the need to extend this picture to a more simply configurable, lightweight DMZ (or better, rings of hierarchical security realms). We see the VO created by federating partially trusting Grids as needing the establishment of short-lived "Virtual DMZs" that may be brought up and down relatively simply. Such systems would be built over the top of more permanent DMZs.

Towards Federation through Mediation

We conclude this report with an overview of secure, federated Grids in practice. As stated in the introduction, we believe that Grids will emerge in a patchwork fashion, building up from small and medium installations that see the need to provide common services to the heterogeneous resources that they own. Such organizational Grids will then be federated into truly inter-organizational Grids on a regular basis when trust (and mitigated consequences of abused trust) is more fully fleshed out.

This report has been primarily concerned with addressing the security implications (and advantages) of federation. The larger issue remains communication across Grid boundaries, especially when the Grids are built from different underlying technologies and use different protocols. Technologies such as JXTA, JINI, RMI, Avaki/Legion and others may be used to build OGSI grids or simply Grid-like systems. We may see this even within Globus Grid installations, where older (pre-OGSI) installations use legacy protocols while newer installations (possibly) move to SOAP. We may also imagine a Grid composed of several different available Grid software pieces. WSDL is protocol independent, so it becomes unclear in the OGSI picture how such protocol negotiation will occur when one Grid component discovers and wishes to access a resource in another Grid in a federation.

Let's examine the requirements of the mediation service from the point of view of Figure 2. Presume that a second, mirror image Grid has been federated with the Grid depicted in Figure 2. Messages between the Grids are exchanged through the Grid analogy to Figure 1. We presume the security mappings needed for cross authentication and account creation have already occurred, and the prospective client has obtained proxy certificates. We shall also presume that Grids A and B are both Globus grids but built with different versions. Grid A uses Globus 2.x and so has several pre-Web service protocols. Grid B is an OGSI implementation that uses SOAP. The client is in Grid B (and so is OGSA enabled).

Federation in this case implies several things besides sharing information through GIIS servers and single sign-on. First, information discovery should be made forward-compatible on the legacy Grid through a mediator/translator running at the Grid boundary. This may be an additional function of the Grid Router GR1. Here GR1 serves as a translator

Second, Grid A may wish to limit its available resources to Grid B clients, allowing them only to access the test bed resources. Here GR1 has the additional task of managing subscriptions to Grid services. All resources in the Grid may be running GRIS/GIIS information services, but it is up to GR1 to check incoming requests to see what subscriptions the outside client possesses. In this case, the client only has a "Test Bed" subscription and so only may interact (through tunneled LDAP queries) with the GRIS/GIIS servers associated with the test bed resources. GR1 itself does not need to maintain the entire subscription list. It may instead distribute this responsibility to the inner Grid Routers.

Third, assuming the client finds what it seeks, it attempts to invoke a process to run an application on the test bed. The problem is that the client is a Web service/OGSA style client that wishes to remotely invoke methods through stubs, while the actual service is accessed through the GRAM protocol and uses the Gatekeeper/Job Manager. The GRs must then serve as translators, exposing an OGSI-style interface for the legacy grid, mapping that interface to, say, a Java CoG client, and translating the incoming SOAP request into a GRAM request. One can generally see a scaling problem in this approach with other "Grid" implementations if there were no OGSI. That is, I would have to have N(N-1) translations to connect N different Grid implementations (a JINI grid to a Globus Grid, a JINI grid again to a JXTA grid, and so on). Here OGSI, because it is language independent, acts as the neutral expression of the interface. If the underlying Grid has implemented its services in OGSI-style WSDL, fine. If not, the GRs act as translators.

For such a scheme to work, however, we will need a bit more information. Particularly, the incoming message from Grid B must provide enough information for GR1 to recognize that it is a Globus 3.x or higher grid with SOAP implementations of protocols. Similarly, one may also get an incoming message for a JXTA-built OGSA grid using JXTA protocols.

Finally, the test bed application on Grid A executes and sends back information to Grid B. The GR intercepts the response, translates it to the appropriate protocol, and forwards it on to the destination.

Again, in the federation scenario, Grid Routers act as publish/subscribe topic management points. By following a hierarchical format, which we believe is a natural organization for VOs, we may construct a system of Grid peers. A particular VO maintains a topic list, together with publishers and subscribers, that is distributed throughout the system—different GRs may maintain portions of the subscription list. In NaradaBrokering terms, there should be a single super-super-…-cluster controller for any particular VO, which may be contacts and searched for available topics. This mega-cluster controller provides a virtual view of the subscriptions and publication rights within its system, delegating the actual maintenance of portions of the list to its child, grandchild, etc, cluster controllers.

References

1. H. Chivers, "Grid Security: Problems and Ideas".

2. M. Surridge, "A Rough Guide to Grid Security." Available from http://eprints.ecs.soton.ac.uk/archive/00007286/01/RoughGuideToGridSecurityV1_1a.pdf.

3. C. Kaufman, R. Perlman, and M. Speciner, "Network Security: Private Communication in a Public World." Prentice Hall (2002).

4. L. Peterson and B. Davie, "Computer Networks: A Systems Approach." Morgan Kaufman (2000).

5. S. Tuecke, et al., "Open Grid Service Infrastructure" Available from http://www.gridforum.org/ogsi-wg/drafts/draft-ggf-ogsi-gridservice-26_2003-03-13.pdf.

6. Several Globus security papers are collected at http://www.globus.org/research/papers.html#Security%20Components.