HotGrid: Graduated Access to Grid-Based Science Gateways

Roy Williams, Conrad Steenberg, and Julian Bunn

California Institute of Technology, Pasadena, California, USA

Abstract. We describe the idea of a Science Gateway, an application-specific task wrapped as a web service, and some examples of these that are being implemented on the US TeraGrid cyberinfrastructure. We also describe HotGrid, a means of providing simple, immediate access to the Grid through one of these gateways, which we hope will broaden the use of the Grid, drawing in a wide community of users. The secondary purpose of HotGrid is to acclimate a science community to the concepts of certificate use. Our system provides these weakly authenticated users with immediate power to use the Grid resources for science, but without the dangerous power of running arbitrary code. We describe the implementation of these Science Gateways with the Clarens secure web server.

1 Science Gateways

For many scientists, the Grid is perceived as difficult to use. There are several reasons for this: accounts are issued in response to a proposal, which may take weeks to review. Security concerns often mean that users cannot use passwords, but rather public-key or certificate systems that have steep learning curves. Users are then faced with just a Unix prompt, which many scientists find daunting. Other challenges include difficult porting of code, arcane archival storage paradigms, the need to learn Globus, Condor, GridFTP, and so on. In return for this difficulty is the promise of access to great computational power, but this only for a small number of dedicated people who understand both Grid and science.

Conscious of these difficulties, we have devised the HotGrid concept reported on in this paper, which makes domain-specific ease of use the driving factor. We believe that the balanced, pragmatic approach outlined will not only provide useful science gateways in the medium to long term, but also strong scientific results in the short term. We will explore the construction of "science gateways" which abstract the architectural complexity of Grid resources and scientific applications implemented on the Grid, presenting their capabilities to communities ("Virtual Organizations") of possibly anonymous scientists, who need no explicit knowledge of the Grid. This approach to science gateways will provide a graduated and documented path for scientists, from anonymous user, to the weakly authenticated *HotGrid* user, through to the power user already comfortable with the Grid.

Typically, a gateway developer is a scientist who wishes to host an application and expose its use with a web form[1][2] – a simple way to describe a computation

P.M.A. Sloot et al. (Eds.): EGC 2005, LNCS 3470, pp. 78–87, 2005.

request, with input fields to allow selection of dataset, parameters, response format, etc., and an output data product returned, for example, as a URL. The developer wants simplicity and transparency: a simple tool to do a simple job. Once the web-form approach shows the service to be useful to a community, there is generally a need to progress to a script-based scheme, or to have the service functions embedded in a familiar data-analysis environment. The gateway is thus utilized in several different ways:

- Web form: the user can use the service in a documented environment optimized for teaching the why and how of the computation that the gateway offers. This interface is also excellent for determining the health and "heartbeat" service status.
- Data Analysis Environment: here the user executes a familiar data analysis program such as ROOT[8], VOStatistics[9], IDL[10], or IRAF[11]. The scientist can concetrate on the scientific goals, rather than the computation logistics. The Science Gateway service is abstracted as a remote computation service, and appears to the scientist just as any other service, except that the considerable potential power of the Grid is employed to execute that service. For weakly authenticated users, only a subset of predefined service commands are allowed, making execution of arbitrary code impossible.
- Programming interface (API): the scientist uses an interface written in a high level language that uses the standard web-service protocols (SOAP, XML-RPC) to access the service. In the case of SOAP automatic generation of the interface stubs can be foreseen thanks to the use of the WSDL (Web Services Description Language) as a service descriptor. The connection of the Data Analysis Environment (mentioned in the previous bullet) can be made using this API.
- Logging in to the Grid itself: the scientist who is a "power user" can log in to the host node of the Science Gateway and run a chosen application directly from the command line. This is the traditional way of using a centralised computing facility, and it requires the highest form of authentication due to its wide scope.

1.1 Graduated Security and HotGrid

The HotGrid scheme aims to make the Grid easy to use, yet powerful. This is achieved by gradating access semantics starting from the level of anonymous user, who can obtain a small allocation of CPU time, to the power user, who can be allocated a large quantity of resources, but who is required to write a resource allocation proposal to a committee, wait for reviews to be obtained, and finally to obtain authentication clearance.

There is a crucial difference between two kinds of authentication: the power to use a lot of computer resources, and the power to run arbitrary code. The former would, for example, allow a malicious person to execute an excessive number of tasks in the Grid. At worse, this would constitute a "denial of service" activity, which an operator or alert user would notice and have stopped.

In contrast, a malicious person able to run arbitrary code can do very serious damage very quickly, for example by deleting files, spoofing, and expanding their authentication and credentials to gain elevated access in the system and other mutually trusted systems.

Thus the HotGrid scheme is focused on allowing weakly authenticated users the ability to use powerful resources, but not allowing the execution of arbitrary code.

1.2 Levels of Authentication

The following list illustrates the graduation from none, through weak and then to strong authentication:

- A Science Gateway service runs a specific code with parameters that control it. The service is exposed via a web form that allows a small job to be executed anonymously and easily. The user might be allocated one CPU-hour in any 24-hour period.
- At the next level is the *HotGrid* certificate. The user completes a form specifying who they are and what they want to do, and obtains a certificate that allows perhaps 30 CPU-hours to be expended on jobs submitted via the Science Gateway. When the HotGrid certificate expires, the user must fill out the form again to obtain a new HotGrid certificate. (The analogy is filling in a form to get a `hotmail.com` account.)
- If the user needs more resources, a strong certificate must be obtained, from a certificate authority that makes proper checks on the user's claimed identity (eg. DOE, NASA, TeraGrid, Verisign etc.). The user will then attach that strong certificate to the HotGrid form submission, and thereby obtain more resources according to grant allocation policies, perhaps 100 CPU-hours in any given month.
- Finally, the user may apply for Grid resources in the usual way, via a proposal submitted for review (for TeraGrid accounts, see [12]) and then be allocated a power-user account, with the associated right to use, say, 1000s of CPU-hours per month. The user may also continue to use the Science Gateway, or can log in with `ssh` and use traditional job submission techniques.

Currently, most Grid projects recognize only the power user. Some projects are beginning to create anonymous gateways on non-Grid resources. We propose to work with TeraGrid management to interpolate between these with the intermediate levels in the list above, including the HotGrid level.

Initially, we will focus on two scientific application areas, high energy physics (HEP) and virtual astronomy. The communities and scientists involved in these fields are distributed across the nation and around the world. Both groups already have substantial TeraGrid allocations and are planning additional requests. Based on experience with HEP and virtual astronomy, we will widen the scope of our work to support other scientific disciplines.

1.3 Certificate Authentication

A certificate is a means of *authentication*, meaning that when it is properly presented, it can *prove* that the presenting entity is who it says it is. Certificates can be used by computers to prove their identity (*"Yes this really is caltech.edu, not a spoof"*), but in this paper we shall think of the certificate as authenticating a human. Note that a certificate does *not* specify an account or allocation of any kind of resources, but rather it is used to prove identity, so that resources owned by the owner or group of owners can be utilized. A certificate generally has a start and an end (or revocation) date, and becomes invalid after the end date.

The X.509 certificate is a digital document; when combined with a secret *passphrase* and presented to a server, it provides proof that the correct secret phrase was typed. This happens without the passphrase being stored or transmitted on an open connection, so it is very difficult to steal passphrases from a remote location. Of course it is possible that the passphrase has been compromised; perhaps the owner told a colleague, or there was spyware on a machine, or a camera watched keystrokes. If not compromised, however, then presentation of the certificate is strong evidence that the person presenting is the same as the person who thought of the passphrase. Therefore, the requirement for authentication is to make sure that a person asking for a certificate to be issued is who they say they are: this is the responsibility of a Certificate Authority (CA).

A strong CA will only issue a certificate when appropriate proof of identity is presented: passport, driver's licence, or referral by a trusted third party. We can imagine a weak CA that issues certificates with little proof. Examples abound in real life: a passport is a strong form identification since it is hard to obtain without strong personal credentials and effort, whereas a membership card at a public library is rather easy to obtain. There is a compromise between the strength of the certificate and the difficulty of obtaining it. Generally, a weak certificate can only be used in a restricted context: a fake library card can only allow an imposter to steal a few books, whereas a fake passport can be used to build a whole new life.

It is easy to set up a CA and issue certificates. So, those who accept certificates always have a list of CAs that they trust, and any new or previously unknown CA must convince a resource provider to accept their certificates, otherwise they are of no use to the certficates' owners. Part of the difficulty of using the major Grid systems (e.g. the US TeraGrid, the European Grid, the UK eScience Grid) is the burdensome procedure of obtaining a sufficiently strong certificate that is accepted by the system.

However, we hope to convince the administrators of these major Grids to accept a certain class of weak certificate: the *HotGrid* certificates. These will be obtainable quickly, simply by completing a form at a special HotGrid CA web site, with name and address and a short abstract or description of the intended purpose of the Grid work planned. Techniques for proving that a human is filling in the form, rather than a bot, will be employed. The HotGrid certificate will

be issued immediately, but its use will be restricted to a particular time period, or for executing a restricted set of tasks, and/or for using a limited amount of resources.

1.4 User Experience

Our target user in the HotGrid project is the domain scientist who knows no more about the Grid than the rhetoric from the colour supplements. This person is not interested in downloading and installing software, nor in writing a proposal. This person is primarily interested in using the power of the Grid to solve a scientific problem. We assume that they have a web browser, an internet connection, and some time to work on the problem.

The user can simply connect to a web server that runs a science gateway, such as those in the examples below. We expect each gateway to offer minimal functionality to the anonymous user, which will allow our user to try the system and become convinced that a little more effort is worthwhile.

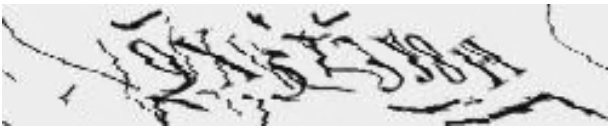

Fig. 1. An example of a CAPTCHA image (Completely Automated Public Turing test to tell Computers and Humans Apart) that can be used to ensure that HotGrid certificates cannot be obtained automatically

At this point, our user will be directed to a CA, and asked to fill out a form with their name, location, and a chosen passphrase. They will also be asked to specify which science gateway they wish to use, and for a short explanation of the envisaged purpose. In the same way as the `hotmail.com` system allocates free email accounts, the HotGrid system will prevent a "bot" from obtaining HotGrid certificates automatically by use of techniques that can differentiate human-based form entries from from computer-generated entries (see Fig. 1). The CA will create an authentication certificate for the user in response to a successful form submission. This will be returned to the user and also installed on the science gateway. We note that this certificate is very weak authentication – in fact all it shows is that a human filled in the form, but we also note that an advantage to the procedure is that domain scientists learn what a certificate is and how it works. Armed with the certificate, our user can now install it in a web browser (online help is provided by all the popular web browsers for this simple task).

With the HotGrid certificate in the browser, the user can now enjoy access to more Grid resources via the science gateway. The certificate is restricted by time and named gateway. If the user wishes to make more extensive use of the science

gateway, it can be done by obtaining a better, longer-lasting certificate, perhaps through referral by a colleague, by membership in a professional organization, or by providing stronger proof of identity. The user may also choose to obtain a power-user account on the Grid system by writing a proposal or by joining an existing trusted project.

Thus, by the use of the HotGrid scheme, we hope to engage and empower a much wider community of domain scientists in Grid-based scientific computing than would otherwise be expected.

2 Clarens

Clarens[13] is a Grid-enabled web services framework that provides a secure, high-performance foundation for constructing web services. It accepts web service requests from clients formatted as SOAP or XML-RPC messages, processes these requests, and returns responses in the same message format.

Clients range from web browsers, through simple scripts, up to large compiled applications. A minimum of software environment requirements are imposed on these clients. Clarens provides a standard application programming interface (API) that is independent of transport protocol or message format.

Two server implementations are currently provided: one using the well-known Apache web server with an embedded Python interpreter, and the other in Java with the Apache Tomcat servlet container. Clarens is agnostic towards operating systems: both Unix and Windows are supported. A set of standard services are available to clients. These include authentication, file access, shell command execution and tools for administration of security. Users with recognized certificates can be organized in a rich hierarchical structure of groups and sub-groups, and thus into virtual organizations (VO).

2.1 Clarens Security

Clients are authenticated using standard X.509 certificates, using SSL encryption, as used by web browsers when communicating with e.g. banking web sites. As an additional precaution clients can also be authenticated using so-called proxy, or temporary, certificates that are valid for a limited period of time, but which still provide cryptographically secure user authentication. These proxy certificates can be generated on the client computer or by making use of an intermediate proxy server that uses a username and password combination for authentication. Once a client is authenticated, Clarens provides access to services, refined by the use of using access control lists (ACLs) for individuals or groups of individuals to each service (e.g. task submission, access to files, permission to modify group definitions, etc.).

Clients are tracked in a session database that stores the certificate details and is able to provide complete session logging for security and debugging purposes. The database is also used by the server to enforce certificate time limits. Other resource limits can be enforced e.g. CPU time, disk usage and network bandwidth usage.

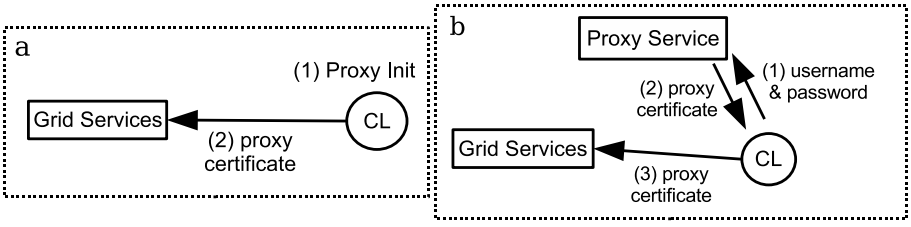

Fig. 2. Two methods of authentication: a) a temporary (proxy) certificate is created on the client machine and presented to Grid services for authentication. b) A proxy service is contacted using a username and password, and a temporary certificate is returned to the client machine. This temporary certificate can be presented to Grid services for authentication as usual

3 Building a Science Gateway

Increased computer security is always a compromise with ease-of-use. The agreement on reasonable levels of security is generally achieved by balancing these opposing requirements. In conventional use of computer centers, there are two main roles: the User and the Administrator. In contrast, when designing and building a science gateway, there are three roles:

- User: this is the person who uses the Gateway, and may be a skilled professional scientist, a schoolchild or anyone with intermediate expertise. This person may be an anonymous user, weakly authenticated or may be a strongly authenticated "power user". It is important to make sure that the former cannot run arbitrary code on the center's computers; and it is just as important not to overly restrict those who have gone to the effort of obtaining a high level of certification.
- Administrator: this person decides on certificate policies, audits code (or trusts the Gateway Developer, see below), and deploys the relevant services so they are accessible and reliable. Furthermore, the Administrator makes sure the Gateway resides in a *sandbox*, meaning that even if the Gateway Developer supplies insecure software, then the damage is limited.
- Gateway Developer: this person (or group of people) has the scientific domain knowledge (what is being computed and why), and also the programming skill to connect domain-specific code into the execution sandbox, and to decide appropriate levels of resource consumption. The Gateway Developer has the responsibility for limiting what the Users can do, and in particular for ensuring that anonymous or HotGrid Users cannot run arbitrary code.

The Administrator can build a sandbox for the gateway by keeping it within a *restricted environment* using the well-known method of partitioning a so-called `chroot` jail, meaning that only files within the sandbox can be read, written, or executed. The Developer, in turn, should ensure that clients with weak authentication cannot run shell commands that are not explicitly allowed for the

purposes of the science domain. The examples in section 4 illustrate these points in the context of real examples.

3.1 Accounting

We have described how a user with weak authentication can be allowed to use the power of a computer center, but without being given the power to run a command that may compromise system integrity. However, the user needs to be restricted further in the sense of resource usage. We do not want a weakly authenticated HotGrid user submitting a thousand jobs to a queue and block the queue with running jobs for weeks.

We envisage HotGrid users obtaining a certificate that has a near revocation date, perhaps one or two days in the future. This will prevent any use of the gateway after that time. Indeed, the user will be unable to access any information about sessions that were started with that certificate.

When a HotGrid certificate is presented, it will be mapped to a generic account name that for all HotGrid users of a particular gateway. These accounts will have an associated responsible "power user", who would generally be the gateway developer. The usage for these accounts will thus be an aggregate of all HotGrid usage.

Usage could also be restricted in a very application-specific fashion. As an example, an astronomical image mosaicker would not allow HotGrid users to create a mosaic larger than a given size. An HEP Monte Carlo event simulator would not allow more than a certain number of events to be generated.

4 Examples

4.1 Astronomical Image Processing

The objective here is a mosaic of astronomical images[14]; an example is in prototype on the Caltech TeraGrid, using the Atlasmaker[15] software. The user provides a string that is the name of an astronomical object or coordinates in the sky, that will be the center of the requested image, together with an angular size of the request. These parameters can be provided by web form or by client API, which sends the request via XML-RPC to the gateway. The user receives by immediate return a monitoring URL that contains the random sessionID. The user can then probe the progress of the of the job through this URL and collect results when the job is finished.

4.2 Monte-Carlo Processing and HEP

Another prototype is a science gateway that allows anonymous and HotGrid users to submit Monte-Carlo jobs that simulate proton-proton collisions in the CERN CMS (Compact Muon Solenoid) detector. Typically, a particular event topology, and a statistically significant number of events are required. The Monte Carlo is a standard piece of the CMS experiment's software stack, and ideally

suited to encapsulation as a gateway service. A form would be completed specifying the desired Monte Carlo generation parameters and number of events. The gateway would take care of parcelling the request into units that could be individually and independently executed on a cluster of computers running e.g. Condor[17], and aggregating the results datasets into a complete file which could then be downloaded by the user.

4.3 Grid Enabled Analysis of HEP Events

The ROOT software is widely used in HEP for the analysis of particle physics data. Akin to tools like Mathematica and Matlab, ROOT offers a rich suite of integrated tools for histogramming, fitting, rendering, annotating and otherwise processing scientific data. While ROOT is commonly run on a user's desktop or laptop, a ROOT server and slave system is available, called PROOF, that allows the event processing to be carried out in parallel on a set of worker computers that are coordinated by a master computer.

The PROOF system is an ideal configuration for an implementation of a HotGrid-based service along the lines described in this paper. A ROOT user, running on a resource-limited desktop system, and thus unable to analyse a sufficient number of events to achieve a statistically significant result, obtains a HotGrid certificate. With the certificate, she authenticates with a Clarens TeraGrid service offering the PROOF system, which then allows her to bring to bear the full power of a PROOF cluster of one hundred TeraGrid CPUs. Moreover, she never needs to leave the ROOT environment: all the dataset I/O and processing are handled in a transparent way, and the impression is simply that her desktop system has suddenly increased in power by at least two orders of magnitude.

4.4 VOStatistics and IRAF

The *VOStatistics* gateway is also in prototype, based on a web service framework developed previously[9]. It provides the client with access to a remote session of the "R" software, which provides powerful statistics and data mining capability. Similarly, the astronomical data processing package IRAF[11] is being made available by the US National Virtual Observatory[16] as a remote service package.

In each of these cases, the user enters a script that is executed by the server, perhaps processing thousands of files. There is, however, a potential danger, because these powerful environments all offer a shell escape command. Therefore we must be careful that we are not providing the ability to execute arbitrary shell commands – by means of restricted shells on the server, as discussed above.

The utility of remote execution is that the remote machine can be much more powerful than the user's workstation, perhaps because it holds a large archive or runs on a multi-processor machine. However, effectively utilizing R or IRAF on a distributed-memory cluster is still a research effort.

References

1. G. Aloisio, M. Cafaro, C. Kesselman, R. Williams: Web access to supercomputing, Computational Science and Engineering 3 (6): 66-72 (2001)
2. G. Aloisio G, M. Cafaro, P. Falabella, R. Williams Grid computing on the web using the globus toolkit, Lecture Notes in Computer Science 1823: 32-40 (2000)
3. Bunn J. and Newman H. Data Intensive Grids for High Energy Physics, in Grid Computing: Making the Global Infrastructure a Reality, edited by Fran Berman, Geoffrey Fox and Tony Hey, March 2003 by Wiley.
4. Conrad D. Steenberg, Eric Aslakson, Julian J. Bunn, Harvey B. Newman, Michael Thomas, Ashiq Anjum, Asif J. Muhammad Web Services and Peer to Peer Networking in a Grid Environment Proceedings of the 8th International Conference on Advanced Technology and Particle Physics, 2004
5. Arshad Ali, Ashiq Anjum, Tahir Azim, Michael Thomas, Conrad Steenberg, Harvey Newman, Julian Bunn, Rizwan Haider, Waqas ur Rehman JClarens: A Java Based Interactive Physics Analysis Environment for Data Intensive Applications Presented at the 2004 IEEE International Conference on Web Services (ICWS 2004), San Diego, July 2004
6. Julian J. Bunn, Harvey B.Newman, Michael Thomas, Conrad Steenberg, Arshad Ali and Ashiq Anjum. Grid Enabled Data Analysis on Handheld Devices Presented at International Networking and Communications Conference 2004(INCC,2004),Lahore,Pakistan.
7. Julian J. Bunn, Harvey B.Newman, Michael Thomas, Conrad Steenberg, Arshad Ali and Ashiq Anjum. Investigating the Role of Handheld devices in the accomplishment of Interactive Grid-Enabled Analysis Environment Proceedings of GCC2003 and Springer LNCS.
8. The ROOT Object Oriented Analysis Framework, `http://root.cern.ch`
9. G. J. Babu, S. G. Djorovski E. Feigelson, , M. J. Graham, A. Mahabal: Statistical Methodology for the National Virtual Observatory, `http://vostat.org`
10. The Interactive Data Language, `http://www.rsinc.com/idl/`
11. The Image Reduction and Analysis Facility, `http://iraf.noao.edu/`
12. Access to the TeraGrid, an overview, `http://www.teragrid.org/userinfo/guide_access.html`
13. Steenberg, C.D., et al.: The Clarens Web Services Architecture. Proc. Comp. in High Energy Physics, La Jolla, **MONT008**, 2003
14. R. D. Williams, Grids and the Virtual Observatory, in "Grid Computing: Making The Global Infrastructure a Reality" by Fran Berman, Anthony J.G. Hey, and Geoffrey Fox, Wiley, 2003, pp 837-858.
15. R. D. Williams, S. G. Djorgovski, M. T. Feldmann, J. C. Jacob: Atlasmaker: A Grid-based Implementation of the Hyperatlas http://arxiv.org/abs/astro-ph/0312196
16. The US National Virtual Observatory `http://www.us-vo.org`, also the International Virtual Observatory Alliance `http://www.ivoa.net`.
17. Thain, D. and Livny, M., Building Reliable Clients and Servers, in *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 2003.