

Usability of the UK e-Science Certification Authority

Jens G Jensen and Matt Viljoen, CCLRC

All Hands Meeting, Sep 2005

Abstract

The e-Science Certification Authority (CA) is a crucial component in the activities of the e-Science community. As one of the largest Grid CAs in the world, we have gained considerable experience with technical and operational issues. However, there are usability issues, some of which are inherent in a Public Key Infrastructure (PKI), others due to the level of assurance the CA must provide to be internationally accepted, others again for various technical reasons. The purpose of this paper is to discuss the most important usability issues, the steps we have taken to alleviate them, as well as the steps we plan to take in the future.

1 Introduction

The Grid Operations Support Centre (GOSC) has been operating a Grid CA covering Grid and e-Science projects in the UK, with the largest community being the UK particle physics Grid community, GridPP. At the time of writing the number of live certificates has grown to 2000 and with over 50 Registration Authorities (RAs) the UK e-Science CA has become one of the largest Grid CAs in the world.

The CA's mission is to provide X.509 certificates for authentication which are acceptable also to international Grid communities, and the CA has been accepted by several large communities, e.g. LCG [3], TeraGrid [5], and EGEE [2]. This is important because most international projects will accept only one CA per country, for various technical and practical reasons which are beyond the scope of this paper.

Since the launch of the CA the GOSC has been providing user support by means of the GOSC Helpdesk. This has provided us with a valuable means of dealing with all problems related to the CA and analysing conceptual problems encountered by users when dealing with the UK e-Science Public Key Infrastructure (PKI). The helpdesk has proved a useful tool in identifying and analysing usability issues which have enabled us to create a roadmap for improving the service offered by the CA and to meet the different needs of users.

The CA software we used before the upgrade was based on OpenCA [4] version 0.8.0; and currently we are using software based on OpenCA version 0.9.2. In both cases, we have made extensive modifications to accom-

modate our workflow as well as both Grid-specific and UK-specific requirements.

In this paper, we discuss the basic usability of the CA, mostly from the novice user's perspective. We discuss things we've already done to improve the usability, and what we have planned for the near future.

2 Background

2.1 CA requirements

In many current Grid infrastructures, security depends on users being able to authenticate themselves to services. More precisely, the Grid security infrastructure is required to solve the following basic problems:

1. Authentication. Each user must authenticate him- or herself to "most" Grid services (some, such as information services, may allow anonymous access). Similarly, each service on the Grid must authenticate itself to the user. This is a basic requirement for all secure connections, to prevent man-in-the-middle attacks.
2. Authorisation. The user's authentication token is used by the service to decide whether the user is allowed to obtain or use a resource managed by the service;
3. Accounting and logging. Abuse of the resources must be tracked back to the user. Accounting is also sometimes used to enforce quotas, or, for commercial services, to bill the user.
4. Each user must generate his/her own key pair. This is an old requirement from the

early days of the Grid PKIs. While there is on-going research in the international Grid CA community to investigate alternative key generation processes, and some progress has been made, this is beyond the scope of this paper.

Thus, the authentication token must be strong enough to uniquely identify the user (or at least allow the identification links to be established), so that authorisation and logging can be secure enough to satisfy the requirements.

The UK e-Science CA provides X.509 certificates for user and host authentication.

2.2 Workflow

For the benefit of readers who are not *au fait* with the workflow from the user's perspective, we provide here a brief summary, slightly simplified:

1. The user uses a browser to open the CA's web page.
2. The user selects his/her local *Registration Authority* (RA) from a list.
3. The browser generates a key pair containing a public and a private key; the private key is stored in the browser and the public key is embedded in a *certificate request* which is stored by the CA.
4. The user goes to the RA to prove his/her identity.
5. The RA approves the request.
6. The CA issues a certificate and notifies the user by email.
7. The user downloads the certificate with the browser that originally generated the keys; the browser matches the certificate with the corresponding private key.

3 Usability

3.1 Basic Problems

Of 820 queries over the last 18 months relating to the CA, we have been able to group the types of problems into the following areas:

3.1.1 Technical problems

Since the launch of the CA the number of novice users has steadily increased and there has been a steady increase in the number of technical problems reported to the helpdesk. Until the recent upgrade, the majority of these problems were due to the lack of scalability of the CA and the lack of support for most recent browsers.

The web server is the principal interface of the CA and an integral component in the certificate request process. As it became apparent that our older OpenCA-based software was incompatible with newer browsers, users were forced to use only "supported" browsers: Microsoft Internet Explorer and Netscape 4.79. Failure to use supported browsers often resulted in unsuccessful certificate requests, or in the case of RAs, a corrupted requests database on the server, leading to loss of certificate requests when we restored from backups. At the same time database scalability issues were becoming apparent — for reasons we have never fully understood, the backend Berkeley Database was causing the signing procedure to take an unacceptably long time.

In 2004 work began in upgrading the CA. The upgrade process completed in May 2005 and has been successful in dealing with the vast majority of the technical problems. Currently all the most popular browsers are supported and the system is considerably more stable, scalable and secure than before the upgrade.

A further advantage of the upgrade is that we have implemented more automated validations in the software. Thus, if users make mistakes, in most cases the interface will catch them. Normally, the RA operator will check the request, and then the CA operator, but in the worst cases, the CA could previously have issued invalid certificates to the users, which will then have to be revoked and the user has to go through the application process again. Another common mistake is that if users use a browser with no cryptography support, such as early versions of Mozilla, the browser will submit what looks like a valid request, but the request contains no valid key, so no certificate can be issued. This mistake is also caught by the new validation processes.

3.1.2 PKI concepts

A more challenging set of problems to address stem from a failure of users to grasp underlying Public Key Infrastructure (PKI) concepts. This often happens with users coming from non-technical sciences. While they don't need to understand the gory details, working effectively with certificates requires a basic understanding of concepts such as the existence and use of public and private keys as well as the mechanics of key generation while requesting a certificate.

A typical recurring problem is users failing to use the same browser to download the certificate as the one they used to request the certificate. The certificate is useless unless it

is matched to the corresponding private key, and the latter is held in the browser which generated the request.

Another example is users not knowing what to do once they have downloaded their new certificate. If they access resources via the browser, there is no problem, but for Grid resources based on the Globus toolkit, as well as for many other uses of certificates, they will need to *export* the certificate and the corresponding private key from the browser, without compromising the protection of the private key.

Yet another example is a difficulty encountered in renewing a certificate. If the certificate and private key are no longer present in the browser, the user will have to *import* them into the browser before they can be renewed.

These problems can partially be addressed by improving the CA documentation and making the documentation more accessible to users. However, we have partly achieved a more effective solution by replacing the browser altogether with alternative ways of interfacing with the CA (see section 4.1.2).

3.1.3 Workflow

In order to provide certificates with a certain level of insurance a very clearly defined workflow exists involving the end user, the RA Operator and the CA Operator. A number of problems that users have sent to the helpdesk have involved a misunderstanding of the workflow, especially what needs to be done in order to request a new certificate. We have addressed these problems by introducing Frequently Asked Questions on the front page of the CA which outlines the steps a user needs to take, as well as more detailed and friendly information at each stage of the request process.

3.2 High end users

Most of our usability discussion so far has focused on the novice user. Now consider the case of the system administrator who has a distributed Grid resource consisting of, say, 100 hosts. Suppose that each of these hosts needs a Grid certificate, because they all run GridFTP servers. The web interface is not suitable for this. There must be a way for bulk request generation, bulk approval, and bulk certificate signing, and bulk download.

In the early days of the CA a bulk request generation script in Perl, and it was later rewritten in bash by Mike Jones from

Manchester University. However, at the moment there is no way for the RA to bulk approve requests, nor can the CA bulk sign requests.

Command line tools have the additional benefit of producing certificates directly in the format required for Grid and other X.509 applications. There is no longer a need for importing and exporting to/from browsers (unless the certificate is needed by in the browser). The disadvantage of command line tools is that they are harder to use by novices and they are less portable to other operating systems.

3.3 Notification

We have found that two simple notifications have helped improving the usability of the CA. When a user requests a certificate, a notification is sent to the user's RA. This is helpful because a few users fail to read the documentation and also fail to read the message on the final page of the request process, that they must go to the RA. In those cases, the RA can help the users by contacting them and helping them through the last stage of the process. It also helps the RA because they can provide a responsive service without having to poll the request repository.

The other case where we have found notification useful is when the certificate has been issued. Because of the security requirements, the signing machine must be unlocked from a safe and the signing key from a separate safe, so the CA issues certificates only once every (working) day. Thus, some users perceive the issuance process as slow. Providing email notification when the certificate is ready can at least alleviate this a bit. Furthermore, the email provides a direct link to the certificate, so it also helps the user download the correct certificate.

3.4 The operators

Although we don't have as many RA operators or CA operators as users (at the moment, respectively, one and two orders of magnitude fewer¹²), it is also important to make the system usable for them. This topic is not the main aim of the paper, so we cover it only briefly.

As explained in section 3.3, notification helps improve the usability for the RA. The largest improvement, however, has been in the upgrade which has allowed the operators to use recent browsers to approve requests. Thus, the workflow for them is easier because

¹In very round numbers 1000 users, 100 RA operators, and 10 CA operators

²Should I delete this footnote?

most people have their favourite browser running on their desktop during their working day.

The other issue, as yet unsolved is the issue of bulk requests, described further in section 3.2.

3.5 Applets

The great advantage of a web/browser based CA is that it can provide a largely platform-neutral, relatively friendly, interface to the CA.

However there are still usability issues with this model. We have addressed some of these by deploying experimental Java applets. In our experiences, the advantage of using applets are:

1. They are (supposed to be) platform neutral.
2. They can guide the users through the workflow just like the web interface does.
3. Applets can perform complicated tasks like the key export/import.
4. Applets can perform validations that the web system cannot, mainly checking the strengths of the passphrase that protects the private key. This is a key advantage to providing the software that runs on the client side. If the passphrase is not strong enough, the applet can educate the user about it. Of course, it doesn't guarantee security forever, because a knowledgeable user can always change the passphrase later, but it certainly helps strengthen the security of the PKI, particularly with novice users.

The disadvantages are mainly related to deployment:

1. The applet has to be signed with a trusted digital certificate to be able to store keys on the user's disk.
2. The applet has to be packaged and distributed with complex cryptographic libraries.
3. Users have to update trust settings in their Java runtime environment. This is particularly complicated because many popular operating systems often have several runtime environments installed, and the user needs to identify the one that is being used by their chosen browser.

With the upgrade completed, we expect to be able to devote more time to support the applets again, and can improve the deployment.

3.6 Policy

The *policy* of the CA, or, more precisely, the *Certificate Policy and Certification Practices Statement*, is a formal document which describes the operational practices of the CA. It is used by resource administrators (or virtual organisations) to decide whether they trust certificates from a specific CA. Thus, the policy document affects the usability of the CA. If the policy is too lax, and users will find that few resources will accept their certificates. If the policy is too strict, users will find it hard to obtain and manage their certificates.

The policy of the e-Science CA was designed to be sufficiently strong that the CA would be accepted in international Grid collaborations.

It is a disadvantage of the policy that it is a rather complicated document, and, as one of the more "mature" in the Grid collaborations, it is getting quite long. It is not feasible for most end users to read and understand the whole document.

One of the complications is that the policy must comply with the law in the country in which the CA operates. For the UK, the next problem is that there is no specific "UK law". Nevertheless, the most important law that the CA needs to address is the Data Protection Act [1].

At the same time, the CA should try not conflict with laws in countries where we wish our certificates to be accepted. We try to achieve this by restricting use of the certificates to the essential for supporting the Grid PKI, namely, authentication.

4 Conclusion

The GOSC runs a Grid CA covering large and diverse communities, ranging from novice users to high-end users. In this paper we have briefly described a few of the usability issues and some of our efforts to improve them.

References

- [1] Data Protection Act, June 2005.
- [2] Enabling grids for E-science. <http://www.eu-egee.org/>, June 2005.
- [3] Large Hadron Collider (LHC) Computing Grid. <http://lcg.web.cern.ch/LCG/>, June 2005.
- [4] OpenCA. <http://www.openca.org/>, June 2005.
- [5] TeraGrid. <http://www.teragrid.org/>, June 2005.