Will client digital certificates ever fly?

A short report from the DCOCE project.

A paper for the UK Unix and Open Systems User Group (UKUUG) meeting, February 2005

Keywords: account creation; authentication credentials; PKI; digital certificates; cryptographic hardware tokens; iKey 3000

Authors: Mark Norman[1], Christian Fernau, Alun Edwards (Oxford University Computing Services).

Abstract

This paper explores the advantages and disadvantages of end user/client digital certificates as means of on-line authentication in a higher or further education information environment. We conclude that the use of client certificates is feasible and scalable. Nevertheless, it is valid to question whether there is a future in such a technology. Certificates could be useful to some users as the front-end authentication tokens for single sign on systems and we believe that it is not critical that most users will never fully understand how they work. With usability feedback from over eighty users, with a broad spectrum of technical abilities, the Digital Certificate Operation in a Complex Environment (DCOCE) project looked further into feasibility issues than most other studies where a common desktop environment does not exist.

Whatever your thoughts about client digital certificates, there is much to learn from the (human) methodologies of public key infrastructure (PKI) and how these can be made to scale. For instance, the approval of an applicant's request for a certificate is carried out by a 'registration authority' (RA). An RA should not necessarily be a technical person, but should be a *trusted* person. If you can trust a person to give out the keys to the building, a membership card or an access pass, then they should be able to verify a user's identity and play a dominant role in establishing user accounts within the organisation. The sysadmins should police the RAs, but the RAs should do the majority of the work: they are the ones that know if the applicant is bogus or for real. This scalable situation should be the goal of account creation mechanisms within large organisations.

Introduction

The Digital Certificate Operation in a Complex Environment (DCOCE) project was a two-year Joint Information Systems Committee (JISC) funded project that completed in December 2004. The main aim of the project was to look into the use of digital certificates by end users in higher and further education (HE/FE) for authentication to services and also to look at the methods of issuing certificates to users and how to manage the 'accounts'. Digital certificates may be used for signing and/or encryption of emails and other documents. These uses of certificates were strictly beyond the scope of the project, but as they are potentially so important to HE/FE institutions, they could not be ignored altogether.

This paper was written assuming that the reader has a basic understanding of public key infrastructure (PKI). For a good summary and background to PKI, please see http://www.dcoce.ox.ac.uk/background/. This resource also outlines some of the challenges to the DCOCE project.

[1] Author for correspondence (mark.norman@oucs.ox.ac.uk)

The project developed most components from scratch and also built up a near 'classical' PKI policy regarding the use of multiple Registration Authorities (RAs) within Oxford University. However, we deviated from a classical PKI design as we made all of the RAs subordinate to a *central RA*. Certificates were issued by GlobalSign (the Certification Authority - or CA) but the architecture of the PKI was such that any commercial or non-commercial (or internal) CA could be used.

How the PKI worked

Policies and practices

The issuing of certificates to staff and students was performed to set procedures (or *certification practices*), as with most registration or account creation tasks. One strength of PKI is that it is accompanied by a legal or pseudo-legal *certification practices statement* that puts constraints upon the procedures that may be used or even details the exact procedures that may be used.

The certificate request/download cycle

In the DCOCE PKI, a user would typically make a browser-based request, before visiting the RA for her organisational unit (OU).[2] The OU is typically a college, department or defined group and the applicant's RA will have means of checking her membership of the OU. Her RA also uses a browser to see that the user has made a request and checks that the applicant is a current member of the OU. After the RA has seen some form of photo ID (usually a university card), he then approves her certificate request. This procedure is outlined in Figure 1 as stage 1 or 2 (with the alternative procedure outlined later). There is a central database where the requests and a little personal information are held. The requests are batched up and submitted to the CA, who generates certificates and returns them to the central database. The applicant either receives an email (if she provided an email address) or merely visits the central database via a web interface after a short time. This allows her to download her certificate. Her certificate is reunited with her *private key*, which was generated at the time of the request (and must remain available or secret to the applicant only).

Once the certificate has been downloaded and installed in her web browser, it is available to be used and to gain access to on-line services.

As mentioned above, there is an alternative certificate-issuing procedure that serves users who visit their RAs *before* making a request from their computer. The users may have to visit their RAs to pick up their university cards and it is unreasonable to insist on two visits. Therefore, the RA checks the applicant's details and issues her with a nine-digit code so that she can make a request from her own computer at a later time. That request is said to have been 'pre-authenticated' and is approved automatically.

(Step 7, as illustrated in Figure 1, is described in *The Local Institution Certificate Store*, below.)

Revocation and renewal

The other important components of PKI are revocation and renewal. RAs monitor the membership of their OUs closely and revoke any certificates associated with any members who have left. The *central RA* may also revoke certificates associated with any kinds of abuse. Renewal of certificates occurs annually as the certificates issued are valid for only one year. (This is because the certificates are authentication certificates and implicitly convey membership of the organisation - Oxford University in our case - and the OU.) Thus, last year's certificate may be used to support an application for a new certificate and no physical

[2] For ease of reading, we use the sexes alternately throughout this document.

re-authentication (i.e. presenting in person to the RA with photo ID) is necessary; however, the RA must be active in the granting of these subsequent requests so that a human check on the continued membership of the users is carried out.

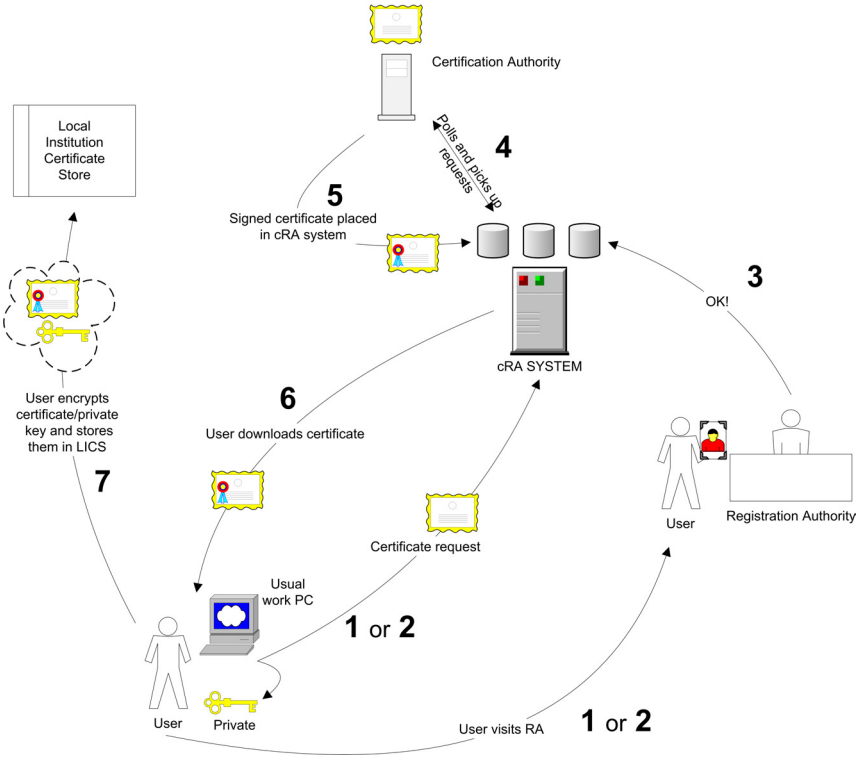

Figure 1 Procedures for requesting and downloading a certificate.

The DCOCE project only trialled revocation procedures to a small degree and was unable to trial the renewal process. However, both were discussed and modelled.

Findings and ideas arising from the design and development phases

Notable differences from other classical PKI implementations

The central RA concept

The DCOCE project used the principle of a two-level hierarchy of RAs. A central RA existed that performed very little physical authentication and the granting of certificates. Indeed, this person or small group was to 'police' the true RAs to ensure procedures were being followed. One of the main duties of the central RA was to recruit and to authorise the RAs and occasionally to grant authorisation status and attributes to end users.

The central RA arrangement is unusual with classical PKI as most PKI texts advocate a hierarchy of CAs [3] rather than a hierarchy of RAs, and we have found very few commercial providers that encourage a hierarchy of RAs. [4] However, in our experience, it seems simpler to leave the complex technical tasks of issuing certificates, revocation lists and managing a

[3] See http://www.microsoft.com/technet/security/topics/identitymanagement/smrtcdcb/sec3/smartc07.mspx#ECAA for an extreme example from Microsoft

[4] But see http://www.diversinet.com/products/passcertauth_benefits.asp for a notable exception from Diversinet Corporation.

mission-critical 24/7 infrastructure to an external CA. The external CA is more able to cope with the task of servicing multiple institutions with multiple certification practices than it is with managing large numbers of RAs in many institutions. The *central RA* approach was taken by the DCOCE project as it became obvious that the scalability bottleneck would be the numbers of RAs across the institution and how close the RAs were to the members of their OUs. If there were only one or two RAs for the whole institution, a classical hierarchy of CAs may be more appropriate, but the registration process would fail hopelessly to scale. (At Oxford University, the different OUs - departments, colleges etc. - are often highly autonomous and it is a difficult challenge to monitor membership centrally. There are people with registration responsibilities in most OUs and it is far more appropriate for these individuals to keep track of membership.)

Pseudonymity

Early in the development phase of the project, we decided to impose an extra requirement upon the design. This was to enable privacy and near-anonymity as far as possible. In a research environment with digital library and database services, some stakeholders thought that we should protect the identity of the individual as much as possible. This is a concept much prized in the digital library sector in the USA and elsewhere, but not very far advanced in such environments in the UK.[5] Nevertheless, it seemed to be a proactive stance and we were deliberately future-proofing our design in case such a requirement became mandatory in future. A complicating factor was our requirement to include the OU on each certificate. This made total anonymity impossible as it becomes obvious to which college or department, within the University, the user belongs. Another complicating factor is that the users hold a single certificate for a year and any unique information on that certificate does not change. Therefore, a service provider may not know exactly *who* is accessing their web-based service but the provider will know:

- they are a genuine member of the University;
- their certificate is valid;
- they have used the service *n* times before (as they recognise the same certificate).

The system that we implemented is based upon *pseudonymity*: a long pseudonym is included in the Common Name within the certificate. This pseudonym does not change but can only be traced to the real user by their own RA or the central RA (i.e. two responsible people within their own institution).

In cases of abuse, a service provider may complain about a particular pseudonym, but only the RA, or central RA, can trace the user.

The Local Institution Certificate Store

One of the usability difficulties for the non-technical end user is the difficulty of moving his certificate if he moves to another computer, or if his usual computer is re-built etc. For this reason, a centralised backup is desirable. However, it goes against some of the principles of PKI that a private key should exist elsewhere, or for it not to be under the exclusive control of its owner. Balancing these considerations, we designed an architecture that we called the Local Institution Certificate Store (LICS). For those users who opted to keep a backup, the LICS contained their certificate and an encrypted version of their private key. The private keys are encrypted by the users, using the pass phrase of their software key store (the 'software security device' in Mozilla/Netscape). This process is mediated by a Java applet delivered via their web browser. This pass phrase is tested for quality but it does not leave the user's PC. If the user needs to access this backup from a computer at a later date, he accesses

[5] Code of Ethics of the American Library Association (1995). http://www.ala.org/alaorg/oif/ethics.html

another Java applet which accepts his pass phrase and sends a hash of this pass phrase to the LICS. A container holding his encrypted private key and certificate is released and the Java applet decrypts the private key and both are installed in the browser.

With regard to security, an attacker who does not know a user's pass phrase would first have to guess the hash (to receive the encrypted private key and certificate) and then to run an off-line attack on the encrypted private key. Since knowing the hash should provide no benefit in guessing the pass phrase and therefore breaking the encryption, this system was deemed to be secure, even against an attack by a rogue sysadmin who would already have access to the 'password' file containing the list of hashes.

Unfortunately, since establishing this design, we have learned of the time-memory trade-off attack, whereby an attacker with access to the list of hashes is able to run an off-line attack and obtain the pass phrase, especially for those private keys encrypted with weaker or more common pass phrases. This attack is dependent upon a fair degree of expertise and a great deal of computing power in order to build a large table of hashes and pass phrases. However, it is likely that these tables will become available on the web with time.

Therefore, it is likely that our LICS is, after all, vulnerable to attack by a determined rogue sysadmin. However, as many authentication systems always implicitly trust their own sysadmins, this is not a great disadvantage and we are still hopeful that a modification of our system entailing salting could solve our problem. Clearly, a determined hacker who obtained the hash 'password' file would also pose a threat, but this can be defended against using the usual defences of good system administration and later detection could result in the revocation of all of the certificates.

In conclusion, our LICS - made use of by the majority of users - is not as secure as we had, at first, hoped. However, it is secure enough to protect the kind of resources that these certificates are to be used against, and more secure than most 'systems' used to protect those kinds of resources at present.

At present, we would prohibit the use of the LICS and no central backup would be taken, were we to issue certificates to protect financially valuable resources (such as accounts databases etc.). Our project aimed to issue *basic level assurance* certificates, which would be inappropriate for highly valuable or sensitive resources and the use of the LICS is quite satisfactory for such a purpose.

Development and user evaluation

The design and development were carried out as will be detailed shortly at http://www.dcoce.ox.ac.uk. Several months' development time was taken by a single developer to build the database, servlets, certificate issuing web site, central database and Java applets.

In September 2004, 39 volunteers attended five 'lab tests' investigating the usability of the DCOCE certificate-issuing interface. Throughout October 2004, more than 80 users tested the interface and the use of the certificates from their own homes/offices. We evaluated the user experience with questionnaires, which recorded mainly qualitative data.

Main findings

PKI related findings

Client usability

In general, the use of the client digital certificates was quite straightforward for most users. However, there were many minor problems regarding usability of the certificate *issuing* mechanism. Nevertheless, there were no 'showstoppers' that could have meant that the widespread use of client certificates was unfeasible in a HE/FE context. The minor problems included:

- different browsers (and different versions) handling certificates in slightly different ways;
- existing versions of the Java SDK requiring upgrading before certificate request/installation;
- problems with the signed Java applet not being trusted (as the Sun root certificate store was usually empty);
- some problems across operating systems (Windows, Linux, MacOS X).

We did not try to educate users in the difficult mathematical basis of public/private keys: something that technically-minded people often need to spend a while working out, before they trust the concept. We found that the ideas of keeping a pass phrase secret *and* of having a 'file' on their computer (or hardware token) were accepted. However, we believe that these two ideas will have to be brought out a little more strongly than our experimental interface suggested for the majority of users to be aware of them.

Once the certificates were installed, the usability reports were overwhelmingly positive.

The user interface for the RAs was simple and understandable for almost all RAs interviewed. The usability issues would not cause any great problem to the scalability of the PKI. However, due to the number of 'minor' issues, it is unlikely that the PKI could be launched on a very *small* scale in an organisation as diverse as the University of Oxford (unless fairly expensive cryptographic tokens were to be used - see *The use of cryptographic hardware tokens*, below). The University 'allows' multiple platforms, operating systems and a wide variety of software, browsers etc. To support this diverse user group would require extensive testing and further development and would give rise to a variety of support issues, meaning that it could be difficult to justify the costs if run on a very small scale.

The use of cryptographic hardware tokens

The DCOCE project evaluated some iKey 3000 USB tokens, originally supplied by Rainbow Technologies and later SafeNet. [6] The tokens included a secure processor running the Starcos SPK 2.3 operating system and 32k of storage. They proved to be implemented easily in Windows 2000 and XP and are supposed to be supported in Linux and MacOS X. Nevertheless, experimentation with these latter two operating systems proved that implementing the iKeys was not a trivial matter, clearly beyond most users and some IT support staff. Other difficult factors emerged, the most notable of which was the fact that tokens initialised in Windows could not be used in Linux/MacOS X and vice versa. However, indications from the manufacturers are that this issue, as well as the difficult implementation in Linux/MacOS X, is likely to be rectified in the near future. It would seem that the open source drivers, developed primarily for Linux, may point to a solution that

[6] See http://www.safenet-inc.com/ (but note that, at the time of writing, the iKey 3000 is not being promoted on the web site).

works well across the operating systems. The manufacturers should also be encouraged to contribute to this effort and/or to provide another solution, although the former option would be far more attractive in the UK HE/FE sector with its myriad of platforms and operating systems.

Despite these non-Windows implementation issues, we believe that the use of the iKey 3000, or similar cryptographic token, is highly attractive to a scalable PKI for HE/FE institutions. There is far less for the end user to do and understand. As long as she has the token in the port and she has supplied (a relatively insecure, and easy to remember!) pass phrase, it all works. At the current levels of technology, the tokens are secure from the attentions of hackers. As well as the easy, intuitive nature of the use of the tokens, the issuing mechanism is far simpler as well. Technical end users - or those requiring high-level assurance - could still carry out a browser-based request but the vast majority of users could be issued tokens that are pre-loaded. This could be carried out on a large scale and is still secure from a rogue sysadmin as the private key never leaves the token: nor can it be read or exported.

The major downside of the use of the iKeys, apart from the - hopefully temporary - cross-operating system issues, is cost. Table 1 shows the likely costs of purchasing the tokens at the time of writing. This is likely to be prohibitively expensive for a HE/FE institution, although it is hoped that these costs will decrease with time.

Table 1 iKey pricing with educational discount [7]

Quantity	List Price (per token)
10-100	£36.00
101-500	£32.40
501-1000	£29.70
1001-2000	£27.90
2000+	£24.00

Operating system or software?

One technical observation regarding the storage of client digital certificates is that, Unix and other systems expect the certificates to be handled only by the software/applications. Windows, in contrast, takes this as an operating system task. We agree that it should be the operating system that stores and protects the certificates and private keys, as - ideally - operating systems should be more secure than applications.[8] This does lead to user confusion in that most users will not understand why they need to 'download their certificate' again if they wish to change browser from Internet Explorer to Mozilla or vice versa.

Usability conclusions

In conclusion, the use of client digital certificates would be a feasible option for an authentication system at a HE/FE institution, but it would be difficult to support on a small scale if the institution had a broad policy of support for multiple platforms, operating systems and software. The use of cryptographic hardware tokens would solve most usability issues and is a very attractive option, but currently expensive. During the certificate application process, some effort should be made to illustrate the concept of something existing on the user's computer that is protected by their pass phrase, and that the authentication system is based upon the user keeping that safe and secret. We believe that this concept is within the

[7] These prices are indicative and do not represent a formal quotation from SafeNet.
[8] We are aware of the irony of this statement, but believe that, in principle, Microsoft Windows takes a better approach in handling private keys in the operating system, rather than leaving it to the applications.

grasp of most users, and it is not necessary to burden the majority of users with ideas of public and private keys.

Findings with relevance to account creation

Although the DCOCE project was primarily concerned with the usability and scalability of PKI at a 'complex' higher or further education organisation, we believe that other authentication and authorisation technologies can learn from the PKI policies/practices of registration. No matter what credential is used - username/password, one time password, digital certificate etc. - the policy of using a trusted individual, close to the applicant, for registration should be universal.

Early in our study, we decided that RAs should be as close to the individual, within his organisational unit, as possible. By making the granting of the application very simple and straightforward, it is practical and desirable for registration to be devolved down as far as possible. Registration should not be carried out centrally or (worse) by the sysadmins. The sysadmins or central registration team - where used - should *police* the RAs to avoid fraudulent and mistaken applications being granted.

In terms of the chain of trust, this is *not* a radical idea. Currently, central registration (e.g. based in 'computing services') *has* to trust a prominent individual in the Chemistry Department in order to create an account or authentication credentials for a new chemistry applicant. Even if another credential is used and trusted - such as a university card - the applicant only holds that card because the trusted individual in Chemistry has granted it to her! The shorter the chain of trust, the better, so it is optional to allow a local trusted individual, such as a personnel officer, or student registration officer to grant the authentication credential and trigger account creation. Anyone with access to current staff lists and/or student applications and attendance information could be an RA, if they hold an existing position of trust within the organisation. The activities of these individuals, and the entries in the database should be monitored by the sysadmins, or central registration teams, who have the skills to do so effectively. These individuals should be able to use or write algorithms to scan logs and cross-check databases.

Thus, the use of many devolved RAs is something that all authentication and account creation methodologies could employ. The central, technical staff is employed to its greatest potential, the chain of trust is shortened and the system should be more accountable in dealing with mistakes and fraud.

Authentication and authorisation

Continuing the argument from the previous section, it is clear that it is very difficult to separate totally the procedures of authentication and authorisation. In our *authentication* project, we *had* to include some authorisation procedures and functionality for users to be able to access the correct services. Figure 2 shows our basic authorisation interface - accessed by the RA - that included further attribute information. The user displayed is a member of the IT Support Staff (ITSS) network and has a university card number of '123456'.

The maintenance of authorisation status and attributes was carried out in our pilot study by the RAs. These trusted individuals were able to, for example, assert that a user was a member of the IT support staff network, and thereafter the certificate holder could access ITSS restricted web pages using only his certificate. However, it is certain that the RA would not have enough knowledge to be able to authorise the user for a wide range of services, some of which would be beyond her knowledge or experience. We established early on that little or no authorisation data should appear on the certificates, as this is prone to change and it would

also erode the 'level' of anonymity. Therefore - as could have been predicted - we concluded that authorisation should be separated from authentication as much as possible.

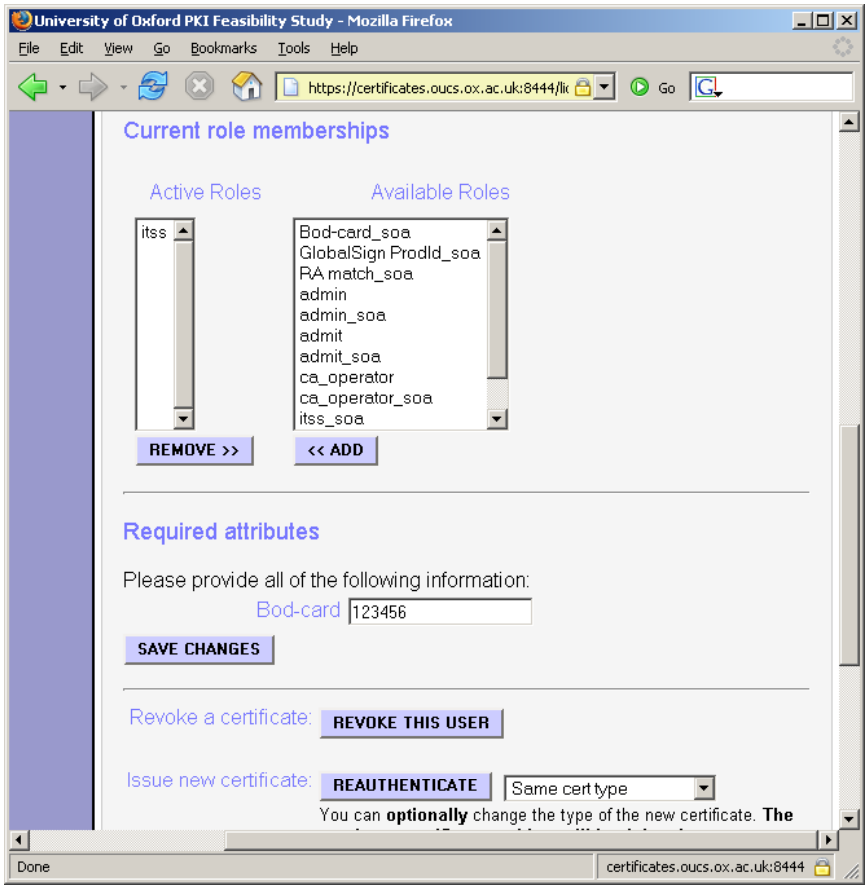

Figure 2 Part of the RA interface that shows an authorisation example

Perhaps unexpectedly, the one area where certificates may form a 'quick win' is related to authorisation. We found that certificates were very attractive to OUs within the University (the *Organisation*) in that, with very little effort, they could be used to allow only members of that OU to access departmental or college web pages. Figure 3 shows a snippet of a configuration file that allows an Apache server running mod_ssl to only allow access to members of Oxford University Computing Services ("oucs"). This is incredibly simple and means that the web site owner does not have to worry about maintaining lists of authorised users. This is a major saving in effort.

```
SSLRequire %{SSL_CLIENT_I_DN_CN} == "GlobalSign PersonalSign
                                     Class 1 CA" \
and    %{SSL_CLIENT_S_DN_O} == "Oxford University" \
and    %{SSL_CLIENT_S_DN_OU} == "oucs"
```

Figure 3 Apache server/mod_ssl code filtering for organisational unit

Is there a future for client digital certificates?

This is an interesting question in that our conclusions should be that, with a little effort from the browser manufacturers, client digital certificates should be usable and they are able to be used in a way that very high security (*high level assurance*) can be attained. However, back in the real world, there are developments such as *single sign on* authentication systems (SSO) and the *Shibboleth* authorisation attribute communication system that may preclude the

growth in client digital certificates as used for authentication.[9] [10] SSO is usually web and username/password based and this has the benefit of familiarity for most users who understand that a successful SSO means fewer passwords to remember. A fully functioning Shibboleth federation may remove the 'quick win' advantage of using certificates (as outlined immediately above) as web site owners and sysadmins will not be concerned with users and authorisation status on an individual basis.

Where client certificates could have an advantage in a world that contains both SSO and Shibboleth is in avoiding the need for separate authentication credentials for accessing services within the home institution and outside the Shibboleth domain. In short, client digital certificates *could* be used instead of username/passwords for SSO systems (or as an optional alternative). If a user can present his digital certificate to a distant service provider who has not signed up to the Shibboleth federation, that distant service provider is able to see that the certificate is a valid and trusted Oxford University certificate and is very likely to trust it. Thus, the user is able to use the same credential to access his home institution's services (in this case Oxford's) as he does to access the remote service provider. Furthermore, were he to access a remote service that was part of his institution's Shibboleth federation, again the same credential could be used. Thus, digital certificates could be seen as facilitating a global SSO experience from the users' viewpoint.[11]

Conclusions

The PKI-related conclusions of our project include the following, that:

- the use of PKI and client certificates *is* feasible and scalable;
- users do not need to understand the esoteric nature of public/private keys - they merely need to understand that there is something that needs to be kept secret, but available on their computer, for the procedure to work;
- cost-effectiveness could become an issue in institutions that support multiple operating systems and software, if relatively few certificates are to be issued (but cryptographic hardware tokens could be used to mitigate for this);
- the use of cryptographic hardware tokens to hold each user's private key and certificate are highly desirable as they ease usability and scalability by a great degree;
- hardware tokens that are cryptographically secure enough are probably too expensive at present and there are currently some operating systems compatibility issues to be resolved;
- the use of client digital certificates can make some common authorisation problems trivial to overcome;
- authentication and authorisation should be separated as much as possible (despite digital certificates being able to accommodate authorisation information within their fields);
- it makes better sense to store authentication digital certificates (and private keys) in the operating system, rather than in software.

Other conclusions from the project include the following, that:

[9] Oxford University is implementing Stanford Webauth as a single sign on system. It can be seen at https://webauth.ox.ac.uk/ and further details may be found at http://www.oucs.ox.ac.uk/webauth/.
[10] This is a poor short-hand description of Shibboleth. Strictly it is neither an authentication or an authorisation system, but facilitates both. See http://shibboleth.internet2.edu/.
[11] SSO is usually interpreted from a technical view point, but most users understand it as "I only need one username/password", rather than "I only need to authenticate once".

- the system of using RAs for account creation triggering/activation and the issuing of authentication tokens is highly desirable where the RAs are based within the same OU (department or subunit etc.) as the applicant;
- central registration staff, and especially sysadmins should not play a direct role in authenticating individuals for accounts and issuing authentication credentials;
- such central staff should, however, police the RAs and the database in order to counteract fraud and mistakes.

And will client digital certificates ever fly?

- the future use of client digital certificates in the HE/FE sector is closely related to the fortunes of SSO initiatives and the Shibboleth development;
- we believe that integrating client digital certificates with SSO provides a solution to bridge any gaps that appear between SSO and Shibboleth, especially for those users accessing services outside of their own institution and/or Shibboleth federation.