Authentication of Internet-Based Distributed Computing Resources in Chemistry

Alan P. Tonge,[†] Henry S. Rzepa,*[,†] and Hiroshi Yoshida[‡]

Department of Chemistry, Imperial College of Science, Technology and Medicine, London SW7 2AY, and Department of Chemistry, Faculty of Science, Hiroshima University, Higashi-Hiroshima 739-8526, Japan

Received October 23, 1998

The evolution of the World Wide Web from a model allowing only the public and open exchange of information on the global Internet via simple HTML-based pages to one involving the additional development of Intranets or Extranets with secure interactive client-server processes is discussed. These client-server processes can be customized on the client browser by the use of embedded applications such as Java Applets that allow the client to communicate with other remote applications or databases via a distributed computing architecture. We show[1] how software authentication using digital object signing procedures in conjunction with X.509 certificates was used to develop three authenticated distributed chemical applications. COS (Chemical Object Store) is a database application based on Java Remote Method Invocation, MoldaNet invokes Java3D to create a molecular visualization tool, and JSpec delivers analytical spectral data. We argue that such authentication of chemical resources on the Internet provides one mechanism for increasing the perception of quality and integrity of molecular information disseminated within the chemical community and creates an architecture for electronic commerce to develop in the molecular science community.

INTRODUCTION

The dominant computing model during the 1970s and 1980s was of a mainframe computer with a centralized CPU located in a special room and providing computing services to local dumb display screens. This model was superseded by the advent of powerful and relatively inexpensive desktop workstations in the late 1980s, which permitted the introduction of the so-called interactive client-server or network distributed computing model. This allowed applications to be split among multiple processors, with actions on one local machine creating processes on a second remote machine, for example a computationally intensive scientific calculation or a database query. The separate widespread deployment of Internet technology using TCP/IP networking protocols in the mid 1990s saw the introduction of another model, involving addressable information pages marked up using HTML (Hypertext Markup Language) and served by a HTTP (Hypertext Transport Protocol) server, the so-called World Wide Web model.[2] The server in this architecture is a networked computer, which can receive HTTP requests from any TCP/IP compliant computer in the world and can return appropriate pages to be accessed and viewed by a suitable Web browser at the remote client.

These two distributed computing and publishing models are now fusing in the late 1990s into a more closely integrated system. Thus Web browsers have developed from being capable of simply displaying static HTML pages to also having the ability to become interactive clients which can request services from a remote server. Much of the impetus for this change is coming from the increasing use of the Internet for electronic commerce and other processes requiring secure transactions and authentication, a feature not provided by the original Web model. Security in this sense requires some form of data protection, which would prevent a third party from gaining unauthorized access to the information. Authentication implies a mechanism for tracing the origin of the original creator of the information or computing resource in a highly trusted manner. Increasingly, one is also seeing the original "few-servers to many-client" Web model evolving into one where almost any computer can also be easily configured to become an information or computing server. This in turn has brought a potential for uncontrolled global document and resource publishing and with it a realization that authentication of the server-side of this model is becoming essential. Such authentication in turn would eventually result in a more trusted electronic publishing and computing model in which a mechanism for establishing the authenticity of published materials can be incorporated.

By 1998, only two major browsers, Netscape Navigator and Microsoft Internet Explorer, were widely deployed on client machines. Each browser had the facility to execute so-called Java applets downloaded from the server. This network computing architecture (NCA) model is often referred to in terms of a thin-client connected to an application server. Browser technology had become perceived as having developed to the point where it has become the universal platform-independent front end for Extranets and corporate Intranets, providing an (almost) standardized interactive interface to remote service requests. In effect, there was a strong push for the NCA to become a new standard for distributed client-server operations. The key technology for implementing this was seen as Java, which is an object-oriented programming language developed from C++ and first released by Sun Microsystems in 1995. Relatively easy to learn and with a range of application libraries, it has rapidly taken off as a platform-agnostic

[†] Imperial College of Science, Technology and Medicine.
[‡] Hiroshima University.

10.1021/ci980336g CCC: $18.00
Published on Web 03/25/1999

Table 1. An Overview of the Evolving Global Network Structures

	Internet	Intranet	Extranet
access	open	private	controlled (authenticated users)
users	public	organization/ corporation	business partnership/ Web commerce
information	general	privileged	shared/privileged

programming language for the Web. Applications written in the Java language compile to neutral bytecode classes, rather than binary machine code, and hence are considered as portable or platform-neutral. These class files can be downloaded from a central server and then run on any system, so long as that system supports a Java Virtual Machine (JVM), an executable which can interpret the bytecode. Browsers such as Netscape Navigator and Microsoft Internet Explorer include a Java Virtual Machine, and a Java applet will therefore run on all machines for which the browser has been implemented. The ability of Web technology to deliver reusable software components, such as applets, means it has become an ideal vehicle for developing distributed computing environments (Table 1).

Internet-Based Distributed Systems. There exist several mechanisms that enable communication between distributed systems using the Web. CGI (Common Gateway Interface) is a specification first introduced with the NCSA HTTP server model in 1993. This enables information to be passed back and forth between a browser and a Web server and represents the simplest middleware for communication between a Web browser and some external program or database. This communication is achieved via a CGI application, a script, or even executable program that sits on the Web server and interprets data fields embedded in a client's HTML input form. The CGI application then spawns a secondary process, a program calculating some property or a database query, before returning the results as a reloaded HTML page. This mechanism assumes there is no permanent connection between the client and server, i.e. it is a stateless mechanism where new processes must be continually created on the server for each request received from the client. Such an asynchronous connection can add a significant load to the server when many client requests have to be processed. A significant limitation of the conventional CGI approach is the requirement for continually reestablishing connections between client and server, which presents a restriction for processes that require authentication, such as those involving electronic commerce or indeed in a teaching environment involving authentication. Various modified CGI schemes have been developed, but none have become standard.

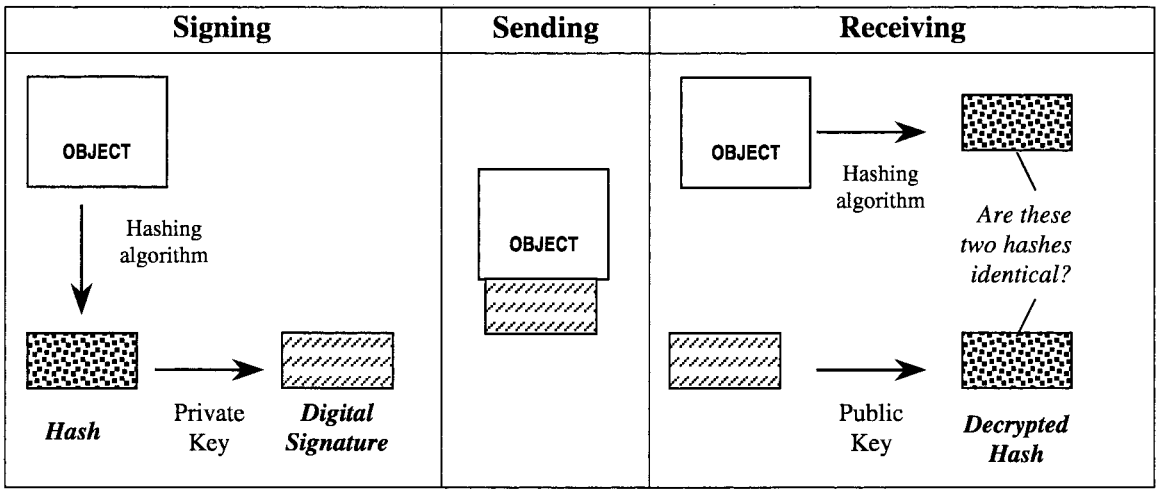
The issue of Web-based authentication and its application to chemical object handling therefore requires a brief introduction to the development of Internet-based object computing. An object is defined as combining data and data-handling methods, with a separate interface to define how other objects are to interact with it, i.e. defines those methods that are available to the client and the parameters required. Objects can reside anywhere on a network, such that a client object does not have to distinguish between local and remote services, as they are assumed transparent, and therefore have the potential to become reusable software components. This model represents a distinct advantage over the traditional procedural methods of programming. Java itself provides methodology that enables clients, principally in the form of applets, to communicate directly with remote systems. The first is the database connectivity specification JDBC, which allows SQL-based operations to be requested of a remote relational database by the client.

The second is a true distributed computing architecture—the Remote Method Invocation (RMI),[3] which permits client objects to invoke methods across a network in remotely defined objects, treating them as if they were local to the client. Object models can be explicitly passed between a server and a client using Java serialization (to convert the object to a byte stream). However, this distributed computing model is not heterogeneous, and it is difficult to use the RMI to access remote objects which have been written in a programming language other than Java, in particular those derived from older legacy systems. Historically, a large number of these legacy systems have been developed within companies and represent a substantial business investment. While industry cannot afford to completely redevelop them in Java, equally neither can it leave them isolated from developments in Web technology. This problem is addressed with a technique known as CORBA (Common Object Request Broker Architecture) object wrapping, which provides an object interface to these systems. CORBA is distributed object architecture originally developed by the Object Management Group OMG, a consortium of industry software developers, prior to the advent of the Web.[3] The central components of CORBA are the object request broker (ORB), which intercepts the call from a client and is responsible for finding an object which can implement the request and the Interface Definition Language (IDL). After defining an object interface, which will allow client and server objects written in different languages to interoperate across a network, the definition is put through an IDL compiler to produce translated output (stubs and skeletons) in the language of choice. These can be used link the client and server objects via the ORB (Object request broker). ORBs are typically commercially written standalone packages, to which potential server objects have been attached/registered and whose functionality is augmented by a range of additional user services. IDLs have been produced for Java, C, and C++. This allows communication either between the ORBs own objects or (via IIOP) objects attached to other ORBs. The combination of Java clients with CORBA-compliant servers. A rather different distributed, competitive but highly proprietary object model has been developed by Microsoft. Known as the Distributed Component Object Model (DCOM),[3] it can support many of the functions devised for CORBA.

The CGI method will survive for some time as a general Internet workhorse, as it requires no special functionality on the client. However, as we move into an era where process and data authentication become more important and perhaps even essential, we anticipate that the three distributed object systems RMI, CORBA, and DCOM will each become increasingly important. Each will provide niche advantages in Intranet and Extranet environments such as complete Java solutions for new developments, compatibility with legacy systems, and major software and operating system compatibility.

Internet Security and Digital Object Signing. The great flexibility of TCP/IP—the communications protocol whereby

Scheme 1. Procedure for Signing and Authenticating Digital Objects



documents are downloaded from server to client in packets via multiple computers over the Internet—means that a document or program could in principle be intercepted in transit and modified by unauthorized persons. One solution to this problem known as Secure Sockets Layer (SSL)[4] operates at the packet level to ensure security, but it does not carry information about the content or nature of the program. Java applets were originally devised as a solution to the issue of interchanging network-transportable programs. Applets were specified to operate inside a so-called security sandbox in order to prevent potential rogue actions on the user's computer system (e.g. reading or deleting files) by a hostile attacker. They are not allowed to make arbitrary network connections, only back to the server from which they were downloaded, and they cannot be used to read from or write files to the local browser machine. These limitations are now considered too restrictive for many distributed computing applications. With the release of Web browsers supporting Version 1.1 or 1.2 of the Java Development Kit, it has now become possible to allow *trusted* applets the ability to work outside the sandbox restrictions.[5] Trusted applets are provided with a verifiable digital signature, authorized by a recognized third party Certification Authority. Both Netscape Navigator and Microsoft Internet Explorer can be used to recognize whether an applet has been digitally signed with a valid object signing certificate (Netscape Object Signing/Microsoft Authenticode).

This solution for authentication of objects (documents or executable programs) sent via the Internet makes use of public-key cryptography, whereby developers get a pair of numeric keys—a public key and a private key—which are mathematically related and may be used to encrypt and decrypt objects. The principle behind the use of these keys is that, provided that they are of sufficient length to inhibit the use of brute force cracking methods, it is not feasible to deduce one from the other. When the Netscape or Microsoft signing tools are used to sign an object such as the class file for an applet which is to be downloaded from a remote server, a one-way hash or message digest of that object, a digital fingerprint, is first created. This consists of a string of numbers of fixed length which cannot be used to deduce the object content and are generally much smaller than the original document. The senders's *private key* is then used to encrypt the hash string to produce a digital signature. A copy of the digital signature is downloaded with the object to the client, who then uses the sender's *public key* to decrypt the digital signature and reproduce the original hash. The user then applies the same hashing algorithm (passed with the digital signature) to the received object to see if it produces an identical hash. The receiving browser then compares the two hashes—if they match then the object has been received unchanged (Scheme 1).

The following section gives specific details of how three particular Java applets were digitally signed[6] and used in this environment, to illustrate these concepts. The first is a client-side database interface component of what we term the Chemical Object Store (COS), which uses Java RMI to communicate to remote object database.[7] A certificate-enabled client is used to acquire chemical data in the form of 3D molecular coordinate files and associated molecular descriptors from the remote source. The second is an example of a distributed molecular visualization and modeling program, MoldaNet,[8] which permits the user to manipulate such molecular coordinate files, and also to read and write the files to a local file store. Such local actions, which are normally disallowed in Java, can now be accomplished within a properly authenticated environment.

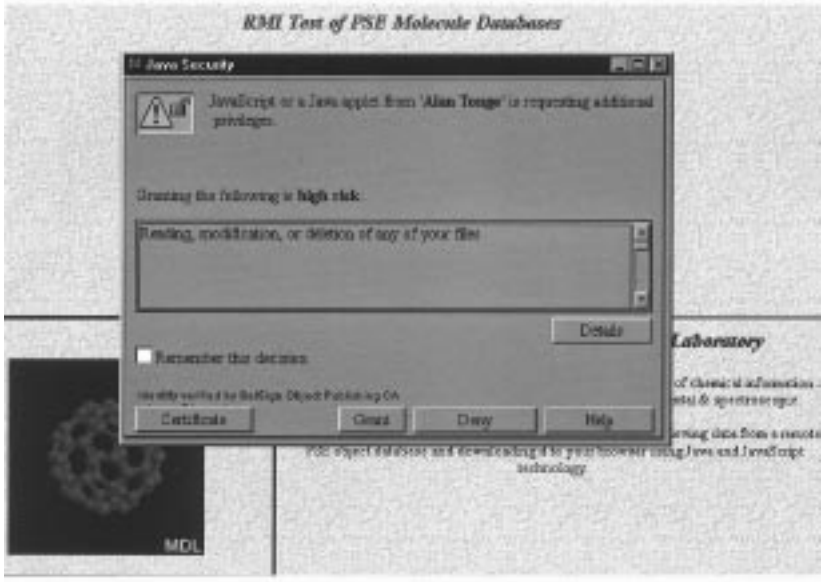
Step 1. Code Modifications to Applet. Only versions 4.0 or higher of both the major commercial browsers have the necessary functionality for recognizing digitally signed applets.[9] Netscape Navigator v4.04 and later requires the additional inclusion of calls to the Netscape Capabilities classes for fine-grained control of granting and enabling permissions to perform activities requiring authentication.[10] Applet source code can be used unmodified with Internet Explorer v4.0. To enable an applet to function with both Netscape and Internet Explorer, further modifications must be made (Scheme 2).

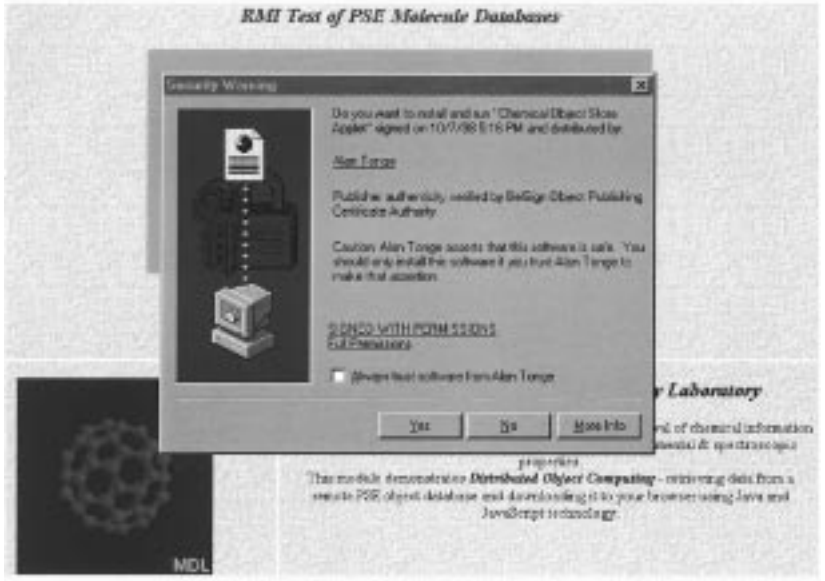
Step 2. Get Digital Certificate from a Certification Authority. There are a number of organizations known as Certification Authorities (CAs) whose own root certificates are either already installed (Netscape) or pointed to (Internet Explorer) on browsers. These organizations will provide two types of service. First, an individual or company can obtain an object signing certificate based on so-called X.509 standards[11] for distributing code over the Internet upon proof of status such as a copy of a driving license or a passport.

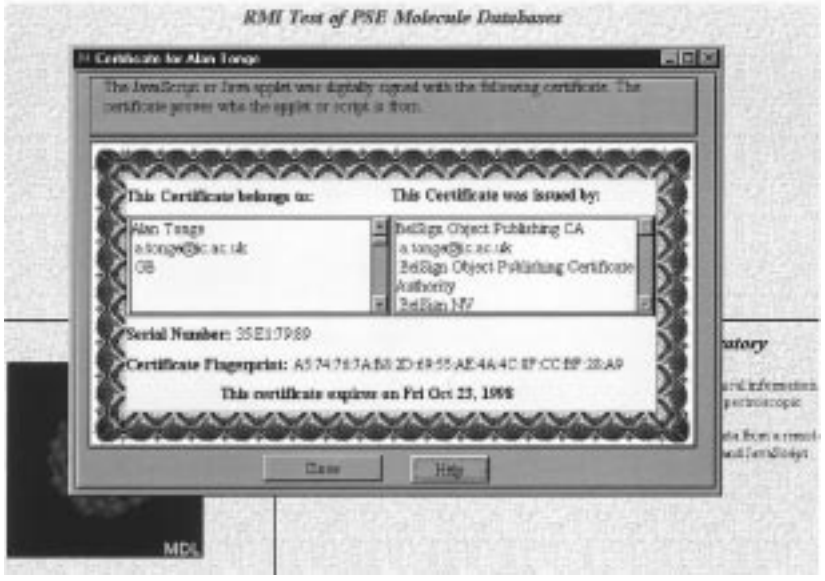
Scheme 2. Example Code To Be Included in a Java Applet To Permit Both Netscape and Internet Explorer Signing

```
import java.applet.*;
import java.awt.*;
import netscape.security.*;

public class myApplet extends Applet {
//    This lists additional Java code that must be inserted in the init()
//    and write() methods in order to enable Netscape security classes.
   boolean hasPrivilege = false, inNavigator=false, inExplorer=false;
  FileDialog openDialog, saveDialog;
  Textarea outText;

  public void init() {
      if ((System.getProperty("java.vendor").indexOf("Netscape")) != -1) {
                inNavigator = true;
      } else if
         ((System.getProperty("java.vendor").indexOf("Microsoft")) != -1) {
                inExplorer = true;
      }
//      MSIE4.0 will give a security exception for FileDialog :
//      com.ms.security.SecurityExceptionEx : FileDialog creation denied
      if (inNavigator) {
//          FileDialog needs a parent Frame
        Component c = this.getParent();
        while (c != null && !(c instanceof Frame)) c = c.getParent();
        openDialog = new FileDialog((Frame)c, "Open PDB", FileDialog.LOAD);
        saveDialog = new FileDialog((Frame)c, "Save PDB", FileDialog.SAVE);
      }
//          Prompt user to enable read/write privileges
      if (inNavigator) try {
            PrivilegeManager.enablePrivilege("UniversalFileAccess");
            hasPrivilege = true;
      } catch (netscape.security.ForbiddenTargetException e) {
            outText.appendText("\nPermission to write files denied ");
      }
  }
  public void printCoords() {
//                Write to new file on client
      if (inNavigator) {
        if (hasPrivilege == false) {
            outText.appendText("\nNecessary write privilege not granted.");
            return;
       }
        PrivilegeManager.enablePrivilege("UniversalFileAccess");
      }
      PrintWriter out;
      String dirName, filName;
      try {
            if (inNavigator) {
              saveDialog.show();
              dirName = saveDialog.getDirectory();
              filName = dirName + saveDialog.getFile();
            }
            out = new PrintWriter(new FileWriter(filName));
//                add out.println() code here to write
//                            to local file
            out.close();
      } catch (Exception e) {
          outText.appendText("\nUnable to write local PDB file. " + e);
      }
//
//          Disable privileges for this method only
//
      if(inNavigator)
            PrivilegeManager.revertPrivilege("UniversalFileWrite");
  }
}
```

Alternatively, the CA can delegate the proof of status requirements to a so-called Registration Authority (RA) which can act on behalf of individuals. Certification authorities currently include Thawte, VeriSign, and GlobalSign (formerly BelSign). We have used GlobalSign for the examples shown here.[12]


RMI Test of PSE Molecule Databases
Java Security
Reading, modification, or deletion of any of your files
Details
Remember this decision
Certificate
Grant
Deny
Help
MDL


RMI Test of PSE Molecule Databases
Yes
No
More Info
MDL


RMI Test of PSE Molecule Databases
This Certificate belongs to:
This Certificate was issued by:
Alan Tonge
OB
BelSign Object Publishing CA
BelSign Object Publishing Certificate Authority
Serial Number:
Certificate Fingerprint:
This certificate expires on Fri Oct 23, 1998
Close
Help
MDL

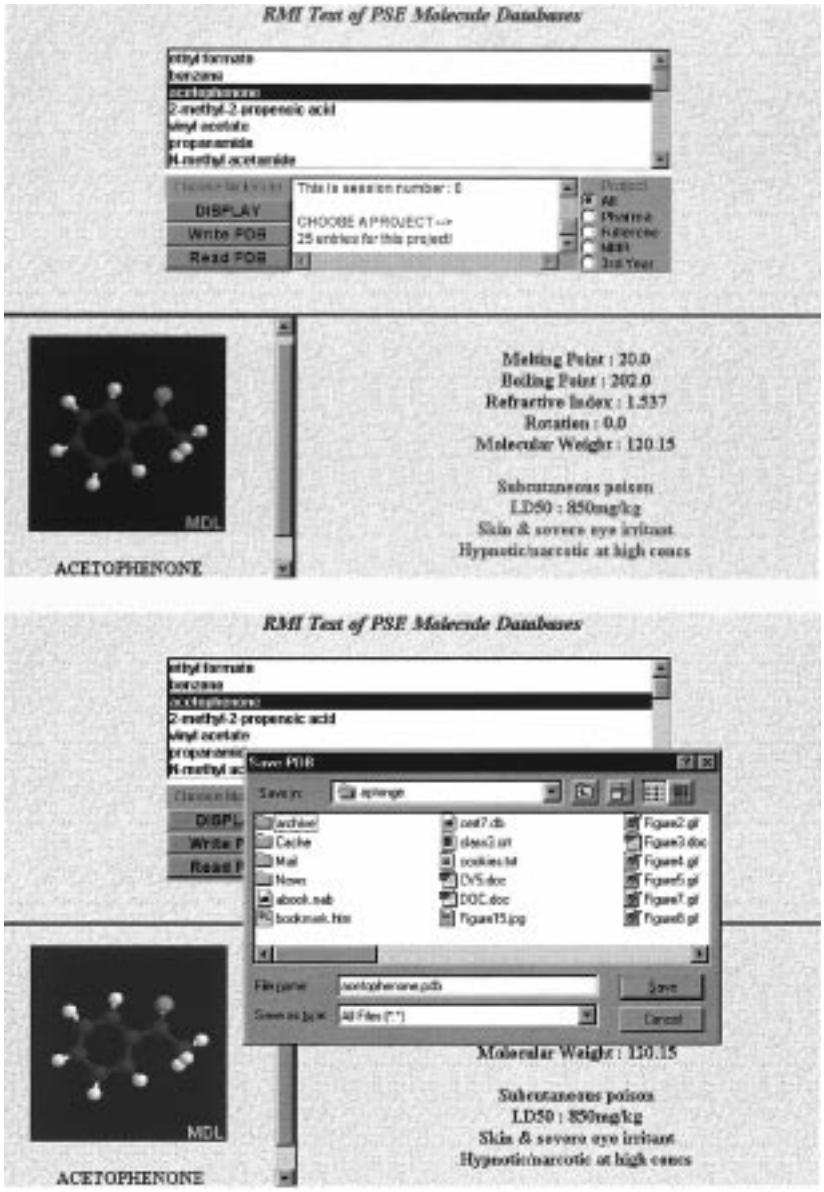


Figure 1. Sequence showing the authentication procedure for a client interface to Chemical Object Store: (a) request to grant security permission requested by applet using Netscape Navigator or (b) Using Internet Explorer, (c) the digital certificate, (d) the COS application in action, and (e) File Dialogs enabled by the granted permission.

To establish the trust which GlobalSign act as a proxy for, we note here that formal documentation describing the terms under which Imperial College was constituted as a University together with documentary evidence of the individuals in whose names the certificates were to be issued was sent by secure delivery to GlobalSign. Certain types of certificates may in addition only be granted by the applicant presenting themselves in person to the Registration Authority. In addition to the object-signing certificates described below, it is also possible for an individual to obtain digital certificates for signing e-mail messages. These personal certificates can also be used to provide that individual with transparently authenticated access to a remote network resource such as an application server or electronic document collection or journal as an alternative to the more conventional account/password combination. From our experiences, we believe this system can be used to establish a level of trust comparable with, e.g. the use of passports for authentication.

Step 3. Sign the Applet Archive. Both Netscape and Microsoft provide software signing tools[13] which allow you to create signed archives of Java class files once you have installed a relevant valid certificate. Microsoft tools can only be run from a DOS prompt on Windows NT/95, while the Netscape tools can also be run on Unix workstations. The archiving procedures differ in that Netscape's *signtool* will create and place the signed applet code in a Java archive (.jar file) whereas Microsoft's *signcode* operates on an existing proprietary cabinet archive (.cab file) created with their *cabarc* tool. Hence, the mechanism for retrieving the appropriate signed applet differs for the two major browsers, although the different tags can be combined in the same piece of HTML code so that each browser can select the appropriate archive:

```
<APPLET CODE="appletRMI.class" ARCHIVE="signed.jar" WIDTH=500
HEIGHT=300>
<PARAM NAME="CABBASE" VALUE="signed.cab">
</APPLET>
```

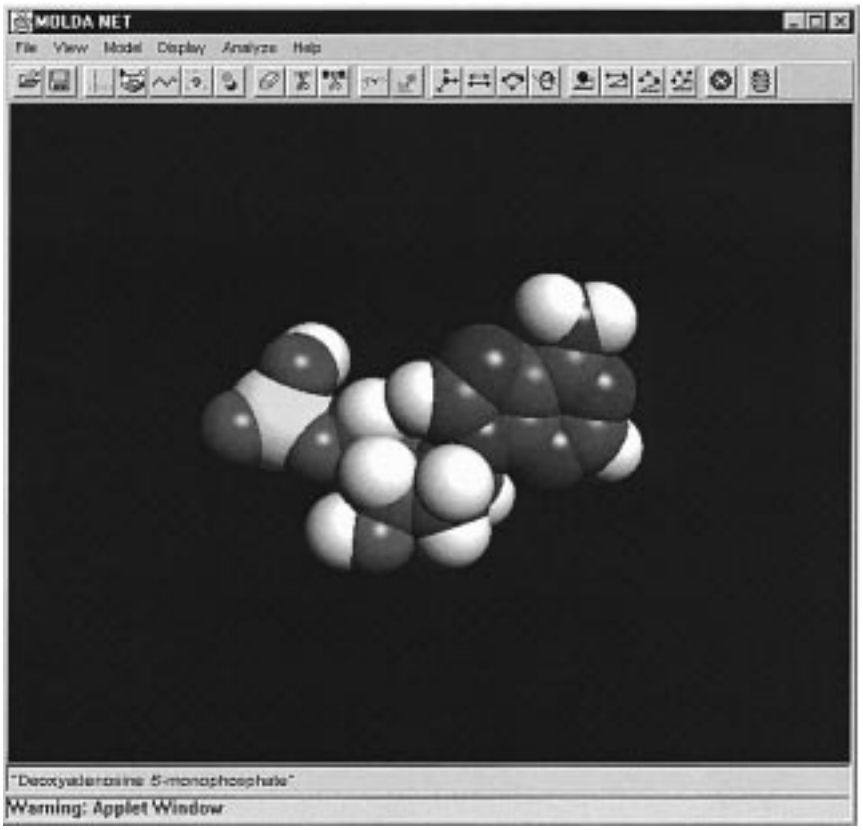


Figure 2. MoldaNet as a distributed authenticated molecular visualization applet. The certificate dialogs are similar to those shown in Figure 1.

Example 1: The Chemical Object Store (COS). COS, as noted above, provides a distributed client object in the form of a Java applet which invokes methods for interrogating a remote object database and returning a chemical data object,[7] which may reformatted into a PDB format coordinate file for display purposes or saving to the local filestore. We believe this to be the first published chemical application of such an authenticated distributed computing technique. For use within a local environment, the user should have access to their local file store in order to save the file. The client should also be able to retrieve files from the local store to be able in turn to submit them to the remote database. This means that the applet client must have access to Java FileDialogue classes to provide this functionality. Such classes are included with the Windows 95 version of the Netscape browser's Java class library, and client access to them is made available if the applet is deemed trusted using the authentication procedures described above. We note that the Macintosh version of the Netscape browser (Version 4.5 was the latest available to us at time of writing) does not yet implement these FileDialog classes. Vexingly, the use of such FileDialog classes appears not to be permitted in Internet Explorer, even with a valid digital certificate.

A demonstration for use with Windows 95/NT Netscape is available from http://origin.ch.ic.ac.uk/vchemlab/cos/signed_applet.html (Figure 1).

Example 2: MoldaNet. Molda was initially developed as a pilot project in creating a molecular modeling and visualization environment and evolved into an implementation using the Java language with the intent of creating a tool for distance learning in chemistry.[8] In its initial form it was created as a conventional application with local file access, and in this form it had to be installed on each local computer as required. As a Java-based application, the user had to also preinstall the appropriate Java run-time environment for their operating system, a relatively nontrivial procedure that might be unduly inhibiting for less expert users wishing to benefit from a distance learning course. By signing the Molda applet as described above, we have been able to create a genuinely distributed chemical application which is acquired by the user as and when needed and has no local preinstallation requirements other than that of the Browser itself. We also note that MoldaNet, as we term the signed network version, also implements the new Java3D class libraries to allow direct access to graphic device drivers to enhance the 3D rendering speeds and some CML (Chemical Markup Language)[14] classes. Together it also permits the user to filter their model into 3D model descriptors such a VRML97[15] or CML for saving on the local disk and illustrates the utility of such a component-based object oriented approach for builidng application solutions.[16] A demonstration for use with Windows and Netscape 4.05 or higher is available at http://origin.ch.ic.ac.uk/mdda (Figure 2).

Example 3. JSpec. This is a simple applet which can be used to read spectral information in JCAMP-DX format[17] from a server and presented to the user for their analysis. Signing the applet with an X.509 certificate enables the user to directly save the spectral information to their local disk. In theory, the reverse process could also be set up, whereby a user with their own personal X.509 certificate could upload their own spectral data to a remote server in an authenticated manner. A demonstration for use with Windows 95/NT and

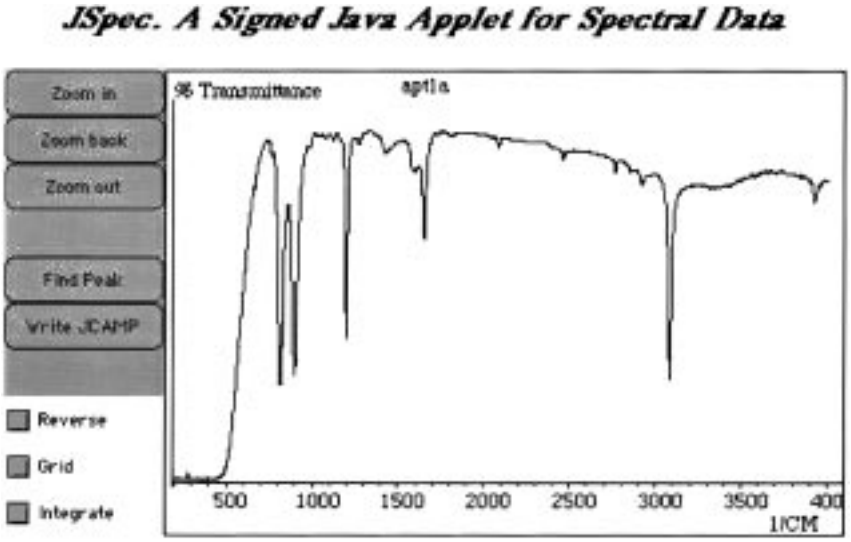


Figure 3. JSpec as a distributed authenticated spectral visualization applet. The certificate dialogs are similar to those shown in Figure 1.

Netscape 4.05 or higher is available from http://origin.ch.ic.ac.uk/jspec-s/ (Figure 3).

CONCLUSIONS

One of the more unusual sociological phenomena of Internet-based interaction during the mid 1990s was that scientific exchange could be conducted with no real awareness of the professional and accredited identity of the participants. The Internet-based Web is moreover now changing rapidly to include not only distributed document delivery services but also the delivery of distributed computational resources. In a chemical context, that might include providing access to e.g. large amounts of data created by combinatorial libraries and high throughput screening and integrated data management across the biological, medical, and chemical disciplines. More particularly, these requirements will increasingly link with electronic commerce and will need to operate within secure and authenticated environments.

In this scenario, one can image the exchange of e.g. molecule objects which could be used to search databases, to start calculations, for inclusion in legal documents such a safety datasheets or for interrogating electronic notebooks or journals. We have presented in this article one model which could potentially be used as the basis for constructing such an environment. This environment includes the ability to create object components, which can be both authenticated and securely signed to create a trusted distributed environment. Whilst the need for such an environment is clearly essential for commercially oriented operations, it is also easy to imagine how it might be applied to the traditionally more open academic and university environments, where so-called identity branding is increasingly recognized as important. Potential areas of application would include virtual courses and tutorials designed for distance learning, where authenticity and accreditation are essential, interaction with and secure submissions to molecule libraries and collections, e-commerce via chemical supplier catalogues, scholarly refereeing of and access to electronic journal articles, and conference organization. We believe that as such models are increasingly adopted, the relative anarchy of the Internet during the period 1993−1998 will be replaced a much higher quality trusted environment of potentially far greater value.

ACKNOWLEDGMENT

We thank the JISC (U.K.) for their award of a JTAP grant, Hiroshima University Venture Business Laboratory for travel funding for HY and Christian Buysschaert of Belsign for help in implementing digital object certificates.

REFERENCES AND NOTES

(1) First presented at the Chemint98 Conference, Irvine, CA, September 12−15, 1998. See also http://www.ch.ic.ac.uk/talks/chemint98/.
(2) Rzepa, H. S. *Internet-based Computational Chemistry Tools*. In *Encyclopaedia of Computational Chemistry*; Wiley: London, 1998.
(3) Plasil, F.; Stal, M. *Software Concepts Tools* **1998**, *19*, 14−28.
(4) Hunt, R. *Computer Commun.* **1998**, *21*, 1107−1123.
(5) Zhang, S. N. *Computer* **1997**, *30*, 76.
(6) Griscom, D. *Code Signing for Java Applets*; http://www.suitable.com/Doc_CodeSigning.html.
(7) Rzepa, H. S.; Tonge, A. P. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 1048−1053.
(8) Ogawa, K.; Yoshida, H.; Suzuki, H. *J. Mol. Graphics* **1984**, *2*(4), 113−116. Yoshida, H.; Matsuura, H. *J. Chem. Software* **1997**, *3*(4), 147−156. Yoshida, H.; Matsuura, H. *J. Chem. Software* **1998**, *4*(3), 81−88.
(9) Users should note that browsers must be correctly configured in order to accept the object signing certificates provided with the downloaded applets. Both Netscape and Microsoft browsers operate a two-stage authentication process, which checks not only the object signing certificate but also the validity of the signing Certification Authority. For this the browser must also have a preinstalled valid CA certificate, otherwise the authentication process will fail. Additionally, all certificates have a limited lifespan, and it is important that the CA certificate is present and not date-expired. Up-to-date root certificates for BelSign are available at: http://www.belsign.be/en/services/receive/install-ca.html.
(10) See http://developer.netscape.com/docs/manuals/signedobj/capabilities/01cap.htm.
(11) Sameshima, Y.; Kirstein, P. T. *Computer Networks ISDN Systems* **1996**, *28*, 513−523.
(12) GlobalSign: http://www.globalsign.net/.
(13) Part of the Microsoft SDK for Java3.1; http://www.microsoft.com/java/download.htm; http://developer.netscape.com/software/signedobj/jarpack.html.
(14) Murray-Rust, P. *Chimia* **1998**, November issue. See, also: http://www.xml-cml.org/.
(15) Casher, O.; Leach, C.; Page, C. S.; Rzepa, H. S. *Chem. Brit.* **1998**, *30*(9), 26. See, also: http://www.chemsoc.org/gateway/molmodel.htm.
(16) For further details of the functionality of the MoldaNet program, see: Yoshida, H.; Rzepa, H. S.; Tonge, A. P. *J. Mol. Graph. Mod.* **1999**, in press.
(17) For further details of this and other standard chemical data exchange formats, see: Rzepa, H. S.; Murray-Rust, P.; Whitaker, B. J. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 976−982. JSpec was written by Guillaume Cottenceau.

CI980336G