Computer Communications Journal, Elsevier, Volume 26, Issue 16, Pages 1863-1872, 15 October 2003 (ISSN: 0140-3664)

Using a privilege management infrastructure for secure web-based e-health applications

B. Blobel[a,*], P. Hoepner[b], R. Joop[b], S. Karnouskos[b], G. Kleinhuis[c], G. Stassinopoulos[d]

[a]*Otto-von-Guericke University of Magdeburg, Medical Faculty, Institute of Biometry and Medical Informatics, Leipziger Strasse 44, 39120 Magdeburg, Germany*
[b]*Fraunhofer Institute FOKUS, Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany*
[c]*KPN Research Groningen, P.O.15000, 9700 CD Groningen, Netherlands*
[d]*NTUA, Heroon Polytechniou 9, Zographou, 15773 Athens, Greece*

Abstract

Within the European HARP project, the HARP Cross Security Platform (HCSP) has been specified to design and to implement trustworthy distributed applications for health over the open Internet enabling both communication and application security services. Certified servlets composed and attributed according to the user's authorisation create certified and signed XML messages. From those messages, user-role-related applets are generated. The HCSP consists of a client environment, web server, an application server, as well as a database server and an archive server. The needed Privilege Management Infrastructure (PMI) has been established by an Attribute Authority and a policy server. The HCSP components are distributed installed over all countries involved. The role-based authorization has been defined according to the policy deploying the user's attribute certificates. The HARP solution has been practically implemented for a Clinical Study demonstrator.

Keywords: Privilege Management Infrastructure; e-Health; Attribute certificates; X.509

1. Introduction

Health applications recording, storing and processing sensitive patient-related information have to meet advanced security and privacy requirements concerning both communication and application security services. This concerns the dimensions of information availability, confidentiality and integrity, but also its accountability and traceability. Because access to health information is legally restricted to the immediate professional needs (need-to-know principle), application security services such as authorization and access control have to reflect the specific user-patient relations, the selection of permitted items as well as resulting rights and duties. Because of the highly dynamically changing relations in a shared care environment, such services can hardly be managed for an individual user. The solution might be the grouping of users according to the user's roles and the grouping of information according to its classification level. This brings the need of all aspects of security to be intimately bound to those roles and players and at the same time be embedded into the medical application. Privilege management enabling access decision and control is based on roles represented by attribute certificates. The attributes are usually bound to a unique identity provided and validated within a PKI. The attribute certificates are provided by a Privilege Management Infrastructure (PMI) [1].

Two technology related conditions make the above functional requirements particularly challenging: (i) the general wish to work over the Web as a totally open communication environment and (ii) the generic embedding of security into the application, so as to prevent particular and repetitious exercises for each individual medical application at hand. So we have to see: (i) medical information as distributed over numerous physical/administrative sites and (ii) medical applications as an ever evolving and expanding suite of facilities to health professionals and the general public.

* Corresponding author. Tel.: +49-37-135-42; fax: +49-67-135-36.
E-mail address: bernd.blobel@mrz.uni-magdeburg.de (B. Blobel).

0140-3664/03/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0140-3664(03)00081-1

HARP[1] [2] was a project within European Union's IST Programme that was initiated to target exactly those demands.

2. Privileges

One of the primary commercial motivations behind the notion of privileges represented by attribute certificates (AC) [1] is the fact that in many e-commerce environments one's attributes are more important than one's identity. Thus for example, on an e-commerce B2B site, entrance may be restricted to those who are members of a certain commercial organisation, or those who have paid a certain membership fee to a professional association. In such a context, the authorisation process is not based on identity per se, but rather on attributes.

The need for attribute certificates has, on the practical level, arisen from the need to use a viable PKI in multi-hierarchy organisations with a wide variety of different authorisation requirements and privileges. Public-key certificates have not proved themselves to be a particularly viable solution in these contexts. Originally, X.509 public-key certificates (PKC) were meant to provide non-forgeable evidence of a person's identity. However, it quickly became evident that in many situations (commercial and other), information about a person's privileges or attributes can be much more important than that of their identity.

This led to extensions of X.509, to enable additional information such as attributes to be kept in a certificate. Notably, X.509 Version 3 introduced the new concept of certificate extensions, i.e. formatted blocks of data that could hold any additional data required. Many systems have taken advantage of this to introduce additional information in private extension fields. However, the somewhat haphazard manner in which this occurred has led to a fragmented system, which lacks some interoperability, jurisdiction, and revocation functionality. Streamlining and standardising this process will be an important part of both extending the use of these certificates and taking better advantage of the current systems.

We close this subsection with some words on the issue of interoperability, which is fundamental to the widespread use of certificates. Though the X.509 standard went a long way in enabling disparate systems to interoperate, the introduction of the certificate extension-formatted blocks led to the situation in which different systems could easily define and implement their own private extensions. This plethora of private extensions without a suitable interoperability mechanism has proved a significant flaw.

The approach of X.509 to the issue of interoperability was a lowest common denominator approach based on the concept of 'criticality'. In this approach, applications that do not understand an extension simply ignore it. This to a large extent eliminates the benefits of the extensions, except in smaller environments where applications can be programmed to mutually recognise extensions. While for organisations using an internal PKI/PMI scheme this can be useful, it does not facilitate the goal of a universal PKI/PMI structure.

Meanwhile, the scene is changing by performing standardisation work on attribute certificates and PMI for health at both ISO and CEN level. Here, ISO 17090 'Health informatics – Public Key Infrastructure' as well as the work items ISO 'Health informatics—Privilege management and access control' and CEN 'Health informatics—Access Control Policy bridging' have to be mentioned, which came to late in the HARP project context, however. If ready, the HARP solution can easily adapted to these emerging standards.

2.1. Attribute certificates

Attribute certificates may convey several different categories of information.

- *Roles*—define the various roles a user may be entitled to perform. These roles are closely linked to the issue of authorisation. Often, they depend on specific attributes and prerequisites such as qualification, education, experiences or skills in general.
- *Groups*—define the user groups to which a user may belong. Grouping criteria may be geographically based, role based, organisationally based, subscription based, etc.
- *Access identities*—provide a means of conveying additional user identification information in the attribute certificate. One example of such is to securely hold a user identity and password, which are required to access a particular system. This class is particularly useful when considering legacy systems and single sign-on applications. By means of access identities, legacy systems can be carried over to an attribute certificate framework.
- *Custom attributes*—a means to specify attributes that do not naturally belong to one of the predefined categories.
- *Restrictions*—a mechanism for rescinding some of the attributes implicitly assigned to a user through membership in a group and so forth.

2.2. Privilege management with attribute certificates

The concept of attribute certificates as described is simple enough. On many levels, however, this relatively simple modification to a standard authentication certificate can facilitate extended functionality that the latter cannot. This is partially due to the fact that as systems expand, the back-office management involved in authorisation of each individual request becomes a daunting task. As such, a single public-key certificate solution becomes an increasingly less attractive option in such a scenario.

[1] HARP (Harmonization For The Security Of Web Technologies And Applications); IST-Project IST-1999-10923.

In Ref. [3], the following analogy has been made to describe the difference between public-key certificates (as issued by today's PKIs) and attribute certificates. A public-key certificate can be considered as a passport —it identifies the owner, it is usually valid for a long period, it is difficult to forge, and it has a strong authentication process to establish the owner's identity. An attribute certificate, on the other hand, may be likened to an entry visa—it is usually issued by a different authority than the passport issuing authority, and it does not have as long a period of validity as a passport. To obtain an entry visa usually requires the applicant to present a passport that authenticates the owner's identity. As such, acquiring the entry visa becomes a simpler procedure. The entry visa will refer to the passport as a part of how that visa specifies the terms under which the passport owner is authorised to enter a country.

The attribute certificate solution thus addresses the jurisdiction issue, and addresses the fact that the most volatile information in a certificate is attribute information (indeed, one changes roles in society and an organisation much more frequently than one changes name). This is done by splitting a certificate into two certificates, one holding identity information and one holding attribute information. The issuing process thus becomes much simpler, and in some cases the revocation problem as well. Indeed, very short-lived attribute certificates (say of one day or even shorter) need never be revoked, since they simply expire.

This approach has many advantages with regards to RBAC (Role Based Access Control). As an aside, we note that while attribute certificates facilitate RBAC, their use is not necessarily restricted to such. There are potentially uses of attribute certificates that fall into certain billing schemes, which do not strictly fall into the framework of RBAC.

There are also challenges with this approach, such as for example how to manage and issue such short-lived certificates. At present there is still debate as to whether short-lived certificates are a better solution than managing longer term certificates and the better choice is often system-dependent.

We now consider some of the specific functional components of attribute certificates such as issuing, distribution, revocation, and identity/attribute relationship.

At present, the basic structure of attribute certificates is defined in X.509v3. A number of standards are being drafted with regards to the more detailed implementation of attribute certificates. Beyond the present work and related items within ETSI, see also the IETF draft for attribute certificates over the Internet [3]. Attribute certificates are handled by a Privilege Management Infrastructure (PMI).

Issuing attribute certificates. The X.509 standard explicitly states that the authority issuing ACs is an Attribute Authority (AA). This might lead to the assumption that the authority issuing ACs will be the same as the one issuing public-key certificates, or at least will be operating under similar rules and procedures. There are however, many reasons why this should not be the case. For one, public-key certificates are often issued by official authorities acting as a Certification Authority (CA). The distance between the end-user and such an organisation is often great. An AC, however, requires a more specialised knowledge of local policies, workflow and corresponding user's rights and privileges. Also, the frequency with which short-lived certificates are issued is not facilitated by the aforementioned CAs. Thus ACs are best issued internally within the domain (organisation, administrative area, catchment area) where they can be issued on a regular and nearly automatic basis. The fundamentally different nature of identity and attribute traits should be reflected in the issuing model that each one uses.

Attribute certificate distribution. There are two primary models for the distribution of attribute certificates. The 'pull' model reflects standards contained in X.509. Attribute Certificates are published in a directory (e.g. X.500) at the time of issue. When an application requires the use of an AC, they retrieve or 'pull' it. On the other hand, the 'push' model involves the user supplying their AC directly to the application at the time of request of access (similar to the manner in which users present a password in conventional access control systems). The choice to use the 'pull' or 'push' method is usually dependent on system requirements and the available infrastructure. Both models have their advantages and disadvantages within certain implementation contexts.

Attribute certificate revocation. X.509 calls for the use of attribute certificate revocation lists (ACRLs) to deal with revoking ACs. This is done in an analogous manner as for public-key certificates. Note that when attribute certificates have a very short life span, it may not be necessary to maintain an ACRL at all. The question of whether to revoke certificates is also system-dependent.

Public-key/attribute relationship. Attribute certificates depend of course on public-key certificates. Typically, the verification process the application performs involves the following steps:

- The application validates the user's public-key certificate, verifying that it is correctly signed by a CA, and by determining that it trusts one of the CAs in the hierarchy under which the certificate was issued. Details of this hierarchy can be found by following the information contained within the certificate.
- The application then verifies the identity of the user, usually by performing a cryptographic challenge-response protocol. The application can verify the responses in this protocol by means of user's public key as provided in their public-key certificate. The nature of the public key/secret key combination assures the application that only the valid party could have provided the correct responses with the challenge-response protocol. These challenge-response protocols cannot be replayed at a later date since the challenge will in each instance be different.

- The application then obtains the attribute certificate, and verifies its identity by determining that it was issued by a trusted entity, that its signature is valid, and that it has not expired. Most importantly, the application checks that the owner of the AC is the same as the owner of the validated public-key certificate. This step is indeed critical, since often attribute certificates are public (and even if they are not explicitly public, in many organisations the nature of a certain employee's authorisations and roles will be common knowledge).
- Lastly, the application checks the attributes within the attribute certificate and then determines whether or not the given user is allowed to access the requested service. There are other variations possible at this stage, such as billing and so forth.

In attribute certificates, as in public-key certificates, the data structure is based on the ASN.1. The issuer information and the serial number from the holder's public-key certificate define the holder information in the AC. When this field is completed, the AC is then bound to the corresponding PKC. As mentioned before, this binding is critical since without it, someone who has been properly identified could still attempt to make use of another party's attributes.

The AC issuer field contains information relating to the AC issuer. In this way it plays a comparable role to that of the issuer field in PKC. Note that the AC time-life is defined in the validity field, and as such separates the privilege life-time from the user information life-time. As mentioned earlier, the lack of distinction between these fields in PKC is a major disadvantage when using PKC to determine access control.

3. The HARP cross-security platform (HCSP)

HARP has provided a generic Platform termed the 'HARP Cross-Security Platform' which is implemented as a set of generic software components (in terms of functionality offered to the user and in terms of embedded and modular security provisions) appropriate and sufficient to instantiate a set of secure medical applications.

The basic components of the HARP Cross-Security Platform as depicted in Fig. 1 are:

- *A client environment*. This is fully under server control and accessible only to players holding the appropriate smartcard.
- *A web server*. The 'entrance'-point for the user.
- *Application (central) server*. User tasks are delegated to servlets; therefore an application server must also host a web server. In our approach, for simplicity reasons, both web and application servers are hosted by the same machine.
- *Policy server*. Policies and policy related functions are provided.
- *Attribute Authority*. The Attribute Authority provides and manages (i.e. issuance, revocation, etc.) attribute certificates.

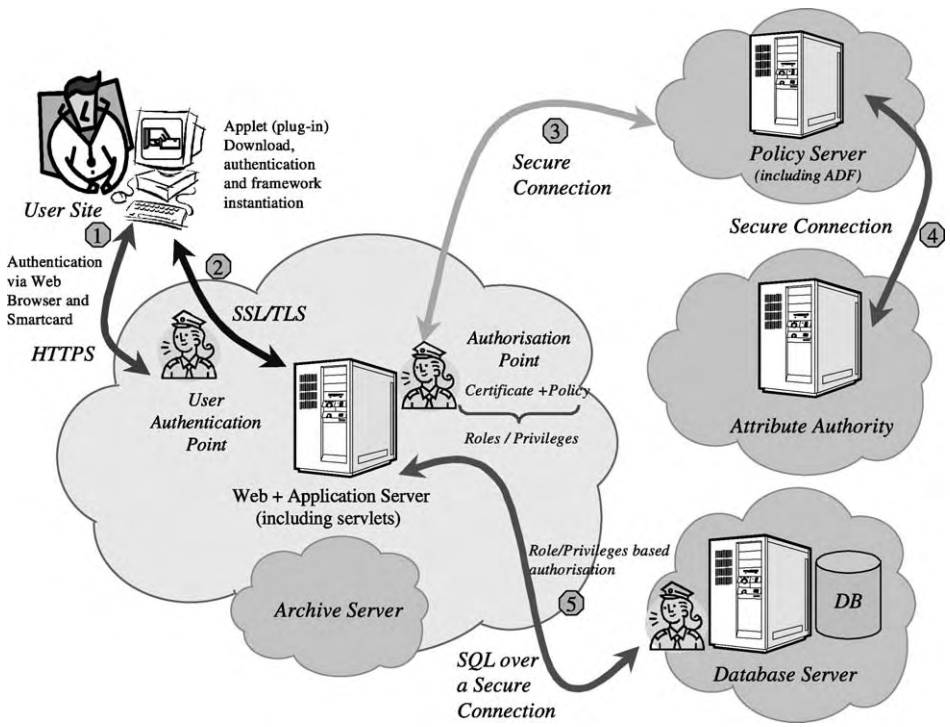

Fig. 1. The HCSP architecture.

- *Database server*. All medical data is stored. Control of access to data is policy-regulated.
- *Archive server*. All messages communicated are stored for accountability reasons.

HARP has adopted a generic scenario with the following properties (step numbering relates to Fig. 1):

- The user connects to a dedicated web server via his browser and uses of course a secure protocol such as HTTPS. (step 1)
- The private key of the user is stored on a smartcard or in a software PSE (Personal Security Environment-prerequisite for the mutual authentication in a SSL/TLS connection). (step 1)
- The web server may accept or deny a connection request based on its policy and the user public-key certificate presented. User and server authenticate with the mutual authentication scheme of the SSL/TLS protocol. The SSL/TLS protocol does not prescribe client authentication in order to establish a secure connection, but the policy defines this (i.e. the web server is configured to request a client certificate). (step 1)
- The web site provides a Java applet execution policy that the user should install on his computer in order to allow the HARP applets to function without problems. This is again up to the site's policy to decide. Finally the applet is automatically downloaded. (step 2)
- The application applet is downloaded to the user's site and further tasks are initialised. The applet initiates a secure connection to the web server in order to take advantage of the available services running within the server in form of servlets. (step 2)
- The identity (ascertained by the public-key certificate) and policy (for accessing data) retrieved from the policy server are used to identify the roles the user is able to take up. This is done via the Authorisation Manager (AM) and depends on the attribute certificates issued and made available by the Attribute Authority. (steps 3, and 4)
- Access to the database server is controlled by the role of the user, e.g. documentation instance, proof instance, student. The database is a relational one. (step 5)
- Correspondingly, on the client side, the presentation view of the application to the user is again controlled by his role; thus presented forms have shaded fields, i.e. fields the user is not allowed to change or see (due to policy) and a set of fields for input/output.

The specific assignment of users to roles mentioned in the previous step uses attribute certificates (in order to certify the role for a specific user identity), which reside in an Attribute Authority. This is the appropriate approach to have the substantiation of roles well demarcated. As a consequence the effect of roles can be clearly separated from the development of the underlying application.

3.1. HCSP client side

Inside the HCSP architecture clients are the part of the platform that provides the interface between the service access points and the end users. Thus HARP provides an open solution, realised by Java applets, that strictly uses Internet technology such as browsers and Java applets enabling the actual functionality the user has been authorised for. After establishing the applet, it sets up a secure communication channel for exchanging information requested from the server or recorded locally.

In general an applet offering services in the concept of HCSP consists of functional components as shown in Fig. 2:

Most of the applet components presented above are custom software developed in the context of the HCSP realisation. However, certain extensions that regard the provision of the XML processing and signing and the establishment of the SSL connections are integrated with the rest of the functionality, offering in this way a complete framework, which can be instantiated in specific medical applications.

The client-side applet was developed using original Sun's Java 2 Development Environment v1.3. Applet downloading and execution works on any modern browser and is handled by the Java Plug-in. In order to be able to deal with issues such as secure sockets, XML parsing, XML signing and smart card reading, other extensions and APIs such as JAXP [4], JSSE [5], XML Security Suite [6], SECUDE [7] are used as well.

Smart card/Health professional card: In the different implementation models for Health Information Systems the Health Professional Card (HPC) and related Trusted Third Party (TTP) services are used within a European security infrastructure and in HARP as well. Hereby, the HPC is defined as a microprocessor card with an additional co-processor specialised for cryptographic algorithms. The certificates provided relate to both

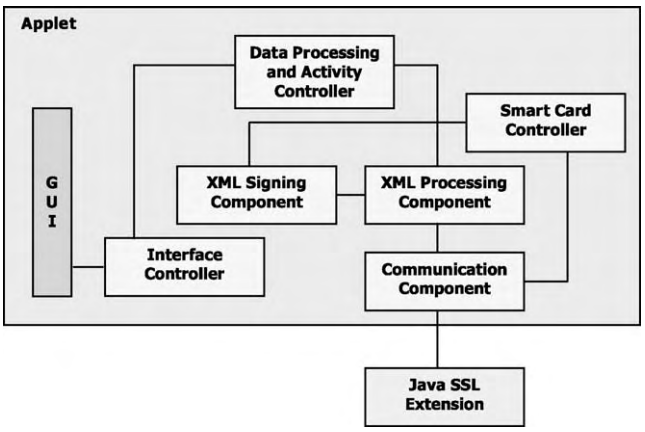

Fig. 2. HCSP applet engine structure.

the identity and the authorisation of the Health Professionals. The identity-related certificate(s) issued by a market-driven and evaluated CA following the new German legislation for both data protection and data security as well as on electronic signatures at all guarantees the first. The latter is expressed by several attribute certificates issued by the Physicians' Chamber (specific domains of care or specific qualifications), by the Statutory Health Care Administration 'Kassenärztliche Vereinigung' (specific permissions), or by employers (hospitals, health insurance companies, etc.). The card contains secret keys with dedicated usage as, e.g. for authentication, digital signature and encryption (e.g. of the session key) as well as the X.509v3-based certificates mentioned. In the card's Master File, the global profession (physician, nurse, etc.) is specified. Beside the authentication, the HPC facilitates also the other communication security services. Based on the identity and the roles of the user on the one hand and the decision rules agreed in the security policy on the other, the HPC also enables the application security services which are related to the person as authorisation, access control, integrity, confidentiality, accountability and audit.

3.2. HCSP server side

In the HCSP the application logic is partly executed in a server environment. A set of functional servers can be differentiated, comprising generic application-related functions as well traditional and enhanced Trusted Third Party services securing communications and applications. Besides the web-/application server providing the servlet execution environment, the Authorisation Manager and Attribute Authority support the generic authorisation and access control functionality for a wide variety of applications. The technical realisation is presented in Fig. 3 and described below.

Web server/servlets: Central parts of the HCSP architecture are the web server and the servlet engine. The servlet

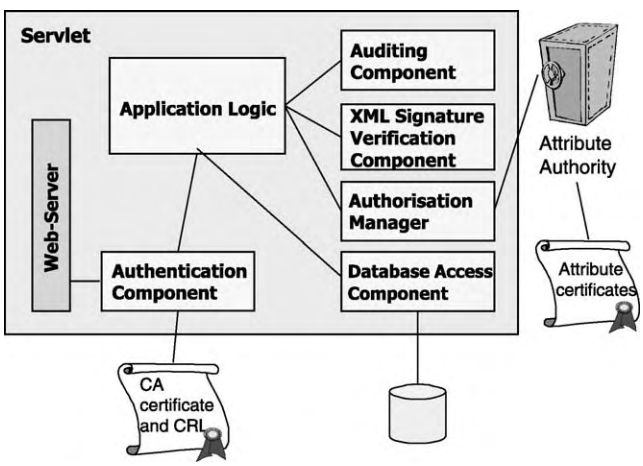

Fig. 3. HCSP servlet engine structure.

implements a service that mediates the data between the users and the database, while they consult the Authorisation Manager for the access decisions. The servlet behaviour is controlled by the *Application Logic*.

From the technology point of view, the servlets run inside the Apache Software Foundation's Tomcat 3.2.1 [8] servlet container. It provides the Servlet API 2.2 (and Java Server Pages API 1.1). An Apache Web Server [9] 1.3.17 provides both static HTML pages and the servlets' dynamic content. Web server and servlet container get connected via the mod_jk. The web server's SSL connections are based on mod_ssl 2.8.0–1.3.17 on top OpenSSL 0.9.6.

Authentication component: The Authentication Component uses the SSL-built-in authentication functionality for client authentication based on public-key certificates. After validation of the certificate the trusted identity is communicated to the Authorisation Manager (AM).

Authorisation Manager (AM): The purpose of the Authorisation Manager (AM) is to provide authorisation and access control services for the application. When activated, the AM uses the trusted identity of the user, the associated attribute certificates and stored access policy information to derive its responses. In a full-scale implementation of the HCSP, an on-line AA is required for returning ACs on request from the AM. The ACs can be generated in advance and stored by (or in connection to) the AC server, or ACs can be generated on the fly containing the requested attributes.

Attribute authority: In HARP the Attribute Authority is off-line located. It distributes issued attribute certificates in a Directory, on-line accessible by LDAP. Upon request the ACs are pulled using OpenLDAP [10]. After verification the AC the attribute information is returned to the Authorisation Manager. The servlet container uses the Blackdown [11] Java environment for Linux. Apache uses the mod_SSL [12] interface to OpenSSL [13] to handle SSL connections from the Debian libapache-mod-ssl package. Our AC generator or Attribute Authority is a custom prototype implementation. Currently it generates ACs with the 'role' attribute. The PKCs, which are needed during AC generation to create the AC holder PKC reference, are imported locally from the file system.

Database access component: The database contains the application data items, such as data about the actors involved (personal and organisational data), further demographics and medical data of patients, as well as their relatives, primary, secondary and foreign keys for organising the database and facilitating data retrieval. The Database Access component enforces the determined access rights based on the attribute certificates. No database-internal access control functions are required. Portability and possible migration to other database systems are thus ensured.

3.3. HCSP attribute certificate visual structure

An access control decision function can retrieve information about the holder of an attribute certificate by examining the AC field 'holder'. It contains a reference to the holder's PKC by means of the PKC's issuer (Certification Authority) and the PKC's serial number. That way the attribute certificate is bound to the corresponding PKC. A client's PKC and AC are both needed for the authentication and authorisation steps to succeed. Both the AC and the AC holder's PKC certification paths must be verified. This means that an AC user like an access control decision function needs at least one PKC and one AC for each client as well as access to the corresponding CA and AA public-key certificates in order to verify the certification paths.

We have chosen to let the AC explicitly reference the PKC which means that clients must be authenticated using a PKC and prove their ownership of the corresponding private key.

The AC attribute certificate fields contain Version, Holder, Issuer, SignatureID, Serial Number, Validity Period, Attributes and signatureValue generated by the Attribute Authority. Figs. 4 and 5 show a client's AC and the corresponding PKC structure.

Authorisation information is placed in the attributes field. It contains a set of attributes that can define group membership and roles among others, as defined by the IETF PKIX AC Profile [3]. All information contained in the attributes field is related to the holder. Our current AC server process generates the 'role' attribute ({id-at-role}) and the roleValues can be selected at run-time.

Extensions can be used to facilitate other security services, typically related to the attribute certificate and not to the holder. Extensions are qualified as critical or non-critical, and any AC user application should be able to read and interpret a critical extension before accepting the certificate. One possible extension for example could be the AC targeting extension ({id-ce-targetInformation}). To target an AC, the target information extension may be used to specify a number of servers/services. The intent is that the AC should only be usable at the specified servers/services. An AC verifier who is not amongst the named servers/services must reject the AC. If this extension is not present, then the AC is not targeted and may be accepted by any server. Extensions are currently not implemented in our AC server process.

Field		Value
version		v1
holder	issuer	CN=Certificate Authority, O=KPN, C=NL
	serialNumber	123456789
issuer		CN=Attribute Authority, O=KPN, C=NL
signature		sha1WithRSAEncryption
serialNumber		6394243734
validityPeriod		December 10, 2000 - December 10, 2001
attributes	type	2.5.4.72 (role) {id-at-role}
	value	doctor
signatureValue		DE54 023A 4933 FFF7 3EDO ED23 …

Fig. 4. Attribute certificate.

Field	Value
version	v3
serialNumber	123456789
signature	md5WithRSAEncryption
issuer	CN=Certificate Authority, O=KPN, C=NL
validityPeriod	September 12, 2001 - December 20, 2002
subject	CN=John Doe, O=KPN, C=NL
SubjectPublicKeyInfo	rsaEncryption
	AAF1 DE54 CA12 34A7 3EDO EC42 …
…	…

Fig. 5. Public-Key certificate.

4. Message sequences in HCSP

For specification of the HCSP semantics and behaviour UML Sequence Diagrams were used. The message sequences for authentication and service selection are outlined below.

4.1. Authentication

The components participating in the 'Authenticate' use case are presented in the following sequence diagram (Fig. 6).

The authentication sequence describes the authentication procedure.

1. *Select URL*: The User selects the URL of the target system (hospital, portal, …) in the browser (Netscape Browser)
2. *https://www…*:The browser connects to the web server. The web server is configured such as to request a client certificate.
3. *Get certificate*:The browser accesses the smartcard of the user to read the user public-key certificate. The user browser/system has to be configured for smartcard access, i.e. PKCS#11, OCF and dll-files have to be installed as required before the system is used (dynamic installations are a future enhancement if required).
4. *Use certificate in SSL*: The browser SSL component transmits the user certificate to the server within the establishment phase of an SSL connection.
5. *Verify certificate*: The certificate is verified within the SSL component of the web server. This might be a local procedure, if all relevant verification information such as the CA certificates and CRL are already available in the Web server or this might be an online verification procedure with, e.g. OCSP to a TTP.
6. *Send certificate*: The X.509 certificate is extracted from the SSL component and handed over to the Authentication Component.

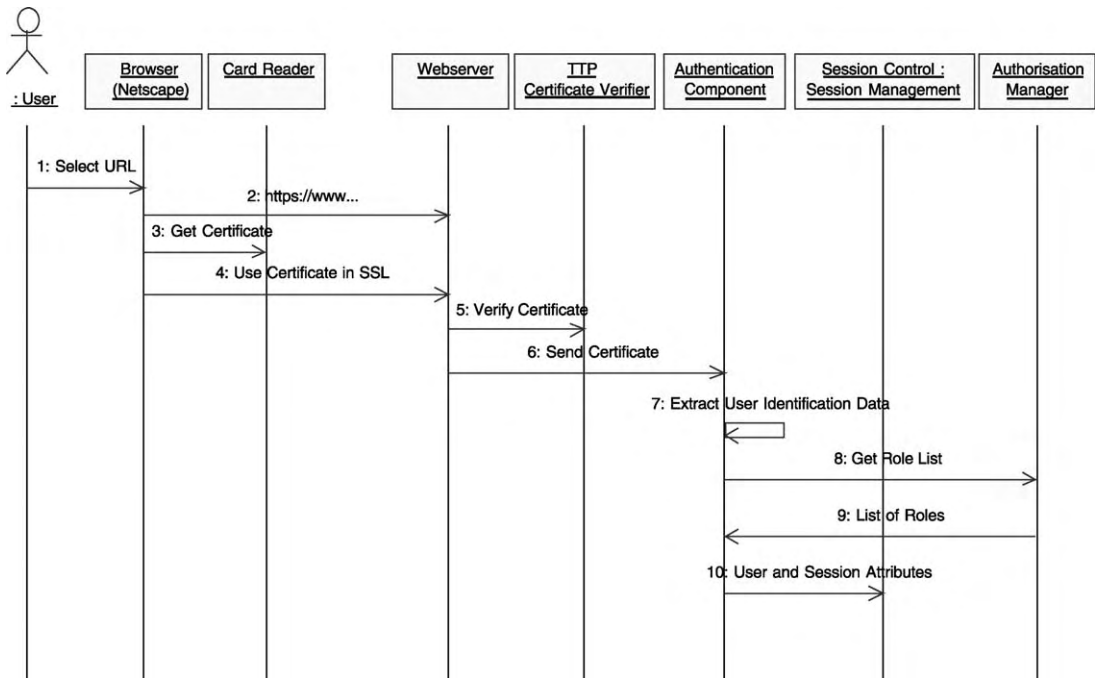

Fig. 6. Authentication sequence.

7. *Extract user identification data*: The unique user identification information is extracted from the certificate. This depends on the authentication policy and can, e.g. be the Distinguished Name (DN) of the user contained in the certificate or the sequential number of this certificate in combination with the certificate issuer information.
8. *Get role list*: The possible roles of the identified user are requested from the Authorisation Manager.
9. *List of roles*: The list of roles is returned to the Authentication Component.
10. *User and session attributes*: Relevant user attributes and session data has to be kept and managed by the Session Control component. Based on these attributes the list of services a user is allowed to access and use may be requested.

4.2. Service selection

Based on the attributes/privileges of the user a certain set of services is available.

The components participating in the service selection use case are presented in the following sequence diagram (Fig. 7).

The service selection sequence describes the selection of a service.

1. *Get service list*: The list of services accessible by the user is requested.
2. *Return service list*: The list of services is returned.
3. *List of services to user*: The list of services is returned to the browser (optional-due to the fact that within a dedicated trial environment only one service is available; an explicit selection by the user is not needed then).
4. *Display list of services*: The browser displays the list of services (optional, see #3).
5. *Select service*: The user selects a service (optional, see #3).
6. *Selected service*: The service selection choice is transmitted to the Session Control component (optional, see #3).
7. *Service access check*: The access to the selected service has to be checked: 'Will user U in circumstances X get access to service S_1?' Based on the user's identity and role, only data and services are presented to the user, which are allowed to be executed. Based on certain policies, this service usage might depend on additional attributes/circumstances such as, e.g. the time of day, the terminal equipment used etc. For simplicity the HARP demonstrator does not take into account these additional attributes).
8. *Request attribute certificate*: A request for available attribute certificate(s) is sent to the Attribute Certificate TTP (optional, see #7).
9. *Return attribute certificate*: The attribute certificate(s) is returned (optional, see #7).
10. *Verify AC*: Possibly a verification of the attribute certificate has to be performed, if not done by the Attribute Certificate TTP already (optional, see #7).

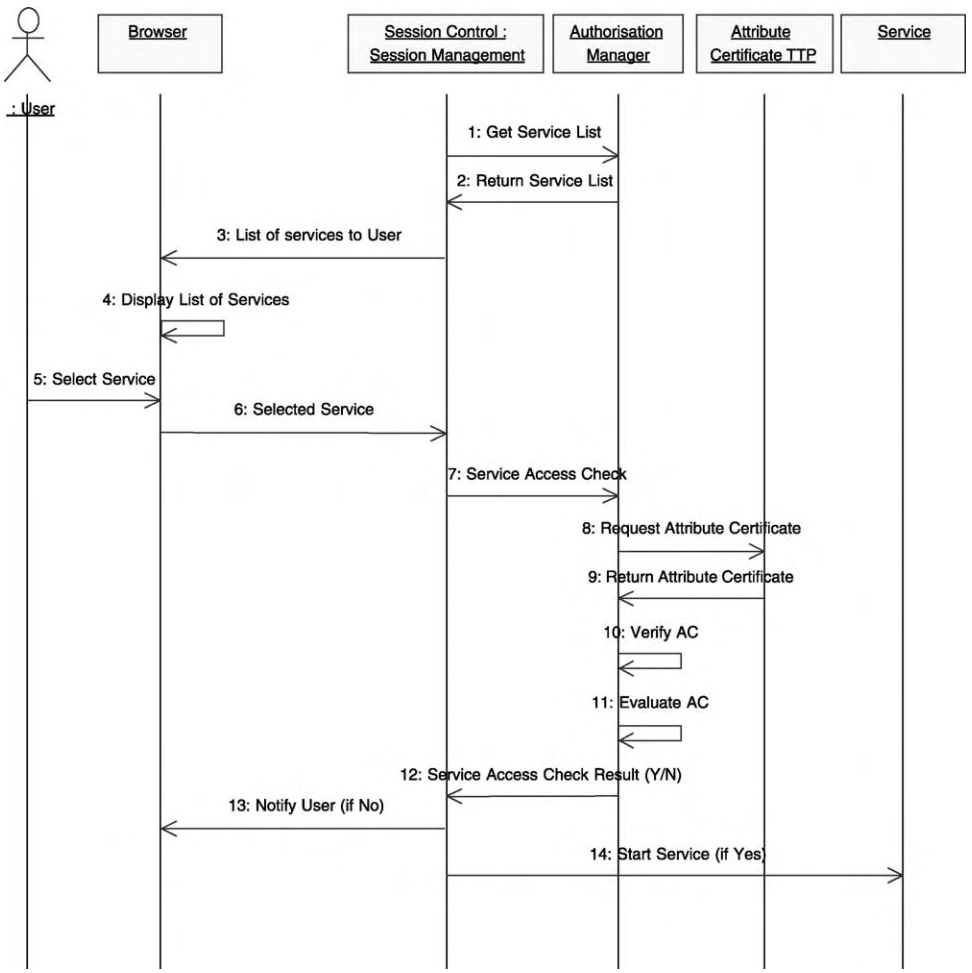

Fig. 7. Service selection sequence.

11. *Evaluate AC*: The attribute certificate is evaluated by the Authorisation Manager (optional, see #7).
12. *Service access check result (Y/N)*: The result of this evaluation (Yes—access allowed or No-access not allowed) is returned to Session Control component (optional, see #7).
13. *Notify user (if No)*: If access is not allowed, the user has to be informed (optional, see #7).
14. *Start service (if Yes)*: If access is allowed, the selected service is started for the user (if the optional sequences are not executed, start service is always initiated if only one service is available).

5. HARP implementation

The described HCSP solution has been practically implemented as a distributed clinical study for quality assurance in pediatrics endocrinology established at the Magdeburg University Hospital. The application enables the setup of the study, its administration, remote data entry, data proof as well as evaluation and deployment of the study. Therefore, the roles of policy council, study administrator, documentation instance, proof instance and study evaluator have been implemented and managed.

To demonstrate the feasibility of pan-European health networks and international clinical studies based on HCSP, the HCSP components have been distributed between the HARP Partners using an established PKI in the Magdeburg region initiated by the Magdeburg Medical Informatics Department.

The HARP approach enables specification, implementation and maintenance of component-based, secure, scaleable, flexible, interoperable applications on the open Internet. Because of the virtual character of the solution, the availability of components and services becomes a crucial factor.

6. Conclusions

The HARP Cross-Security Platform (HCSP) uses a Public Key Infrastructure for authentication and a prototypical Privilege Management Infrastructure for authorisation and access control.

The HCSP consists of a generic applet acting as a user agent and able to produce the GUI as driven by a respective XML message document sent by the server. Moreover the possibilities offered to the user follow policy related access control as embedded into the document; these features are implemented into the applet functionality. In terms of security, the client uses strong mutual authentication with the server and signs all XML messages sent to the server. All cryptographic algorithms and certificates related to authentication, authorization and access control are performed by, or stored on, a smart card.

The HCSP server software is realised as a servlet able to sign and verify XML messages securely exchanged with the client and to map corresponding elements to database queries. Access to the server side database with medical data has also been realised. Access to the database is consistent to the security attributes contained in the XML structure. Access rights are determined by calling an externally provided service on top of and without any reliance on access control mechanisms of the particular or of any future database.

The external service providing access attributes has been implemented. The attributes are compatible to roles that are again connected to the user as identified through the smart card inserted at the client terminal. The server accesses this service and correspondingly (i) embeds into the basic XML document the access control information as security related attributes and (ii) enables/disables corresponding database queries. The mechanism implements the basic principle of collecting (patient-related) medical information according to a policy defined by a policy council (advisory council, ethical commission, etc.) and is a cornerstone of HARP's intention to embed security into the application.

An integration of the smart card into the client environment has been achieved. The implementation is based on SECUDE.

Embedding security into any application to be instantiated over the web-based environment outlined above is based on object oriented programming principles. By associating role profiles and security attributes to standard web based interactions, HARP provides one initial degree of 'automation' in building secure medical applications over the web. Moreover it clearly separates and demarcates security and policy related issues. This enables administrative bodies acting as 'policy councils' to define offline and according to the standing legislation all procedural regulations without entering into implementation details.

Acknowledgements

The authors are in debt to the European Commission for funding as well as to the HARP project partners for kind cooperation.

References

[1] ITU-T X.509, Information technology—Open systems interconnection—The directory: public-key and attribute certificate frameworks, (03/2000).
[2] HArmonization for the secuRity of the web technologies and applications (HARP), IST-1999-10923, http://www.telecom.ece.ntua.gr/~HARP/HARP/HARP.htm.
[3] S. Farrell, R. Housley, An Internet Attribute Certificate Profile for Authorization, Internet-draft June 8, 2001 http://www.imc.org/draft-ietf-pkix-ac509prof.
[4] Sun Microsystems Inc., Java™ API for XML Processing ('JAXP'), http://java.sun.com/xml/jaxp/index.html.
[5] Sun Microsystems Inc., Java(Secure Socket Extension (JSSE), http://java.sun.com/products/jsse/.
[6] IBM alphaworks, XML Security Suite, http://www.alphaworks.ibm.com/tech/xmlsecuritysuite.
[7] SECUDE, SECUDE PKI-Solutions, http://www.secude.de/.
[8] Jakarta Tomcat servlet container, http://jakarta.apache.org.
[9] Apache Web Server, http://www.apache.org.
[10] OpenLDAP, Lightweight Directory Access Protocol (LDAP), http://www.openldap.org.
[11] Blackdown Java environment, http://www.blackdown.org.
[12] The mod_ssl Project, http://www.modssl.org.
[13] The OpenSSL Project, http://www.openssl.org/.