Technology and Health Care 13 (2005) 511–526
IOS Press

PKI-based secure mobile access to electronic health services and data

G. Kambourakis, I. Maglogiannis* and A. Rouskas
Department of Information and Communication Systems Engineering, University of the Aegean, Karlovassi, Samos GR-83200, Greece

Received 27 May 2005
Accepted 4 August 2005

Abstract. Recent research works examine the potential employment of public-key cryptography schemes in e-health environments. In such systems, where a Public Key Infrastructure (PKI) is established beforehand, Attribute Certificates (ACs) and public key enabled protocols like TLS, can provide the appropriate mechanisms to effectively support authentication, authorization and confidentiality services. In other words, mutual *trust* and secure communications between all the stakeholders, namely physicians, patients and e-health service providers, can be successfully established and maintained. Furthermore, as the recently introduced mobile devices with access to computer-based patient record systems are expanding, the need of physicians and nurses to interact increasingly with such systems arises. Considering public key infrastructure requirements for mobile online health networks, this paper discusses the potential use of Attribute Certificates (ACs) in an anticipated trust model. Typical trust interactions among doctors, patients and e-health providers are presented, indicating that resourceful security mechanisms and trust control can be obtained and implemented. The application of attribute certificates to support medical mobile service provision along with the utilization of the *de-facto* TLS protocol to offer competent confidentiality and authorization services is also presented and evaluated through experimentation, using both the 802.11 WLAN and General Packet Radio Service (GPRS) networks.

Keywords: Mobile health records, WLAN, GPRS, mobile healthcare environments, PKI, attribute certificates, TLS

1. Introduction

The healthcare industry continues to seek the ideal computing platform to serve caregivers. The computer-based patient record system expands to support more clinical activities and healthcare organizations are asking physicians and nurses to interact increasingly with computer systems to perform their duties. Existing systems suffer from a number of shortcomings including lack of mobility, bulky obtrusive hardware and lack of flexible functionality. Personal Digital Assistants (PDAs) and Tablet PCs (TPCs) represent a new category of devices, starting to be commercially deployed on healthcare and seem to better match the caregiver's need by adopting the mobile paradigm.

Several groups are working at research level on mobile systems providing access to medical data mostly in terms of telemedicine and remote patient telemonitoring [1–11]. In such a mobile architecture, the sensitive and private nature of the medical information renders security in communications among healthcare providers and patients a critical component. However, although a lot of work exists regarding

*Corresponding author. Tel.: +30 210 2112251; Fax: +30 210 2112521; E-mail: imaglo@aegean.gr.

security (in terms of confidentiality, integrity, non-repudiation of receipt/origin, availability) for Intranet (i.e. Hospital Information Systems) or Internet (i.e. Web Based Telemedicine and Health Record) applications, very few have been done in the field of mobile e-health applications set-up and implementation. The security issues are, in most of these cases, disregarded or not sufficiently handled.

On the other hand, Public-Key cryptography is uniquely advocated to meet the essential security requirements for several sectors, healthcare being one of them, and it has become the preferred means for providing these capabilities. The scope of the paper is to illustrate how Public Key Infrastructure (PKI), Attribute Certificates (ACs) and public key enabled protocols can provide the appropriate framework to effectively support authentication, authorization and confidentiality services. Typical trust interactions between physicians, nurses, patients, hospital administration and e-health service providers are presented, demonstrating that robust security mechanisms and effective trust control can be obtained and implemented. The application of ACs to support mobile service provisioning is also presented and evaluated in terms of service times and implementation constraints, through various experimental scenarios, using the General Packet Radio Service (GPRS) and 802.11b networks, accordingly. In addition, to further prove the feasibility of the proposed schema, the integration of Transport Layer Security (TLS), the well known protocol from the wired networks, and ACs to support mobile e-health transactions and sustain high level confidentiality services to the involved parties, is investigated through extensive experimentation. The measurement results showed that both AC issuing for mobile healthcare applications and TLS protected transfer of medical data are attainable in the context of the anticipated trust model, while at the same time they provide the flexibility and scalability that were designed for.

The rest of this paper is organized as follows: Section 2 discusses some introductory issues concerning PKI and digital certification technologies focusing on ACs' authorization and their competitive advantages over traditional authorization models. Section 3 describes and analyses the proposed architecture and its subsystems, while Section 4 presents an application scenario based on AC authorization that enables mobile access to medical data. Next two sections contribute on the feasibility of the proposed trust model providing experimentation procedures and the conducted results from both AC acquisition and medical data TLS-secured transfer, considering hypothetical deployment in 802.11b and GPRS networks. The last section concludes the paper and gives pointers to future work.

2. Authorization mechanisms, attribute certificates and transport layer security

A Public Key Infrastructure (PKI) [12,13] can be defined as the combination of standard protocols and software that support digital certificates and whose services are implemented and delivered using public-key concepts and techniques. In public key cryptography, each party (user, network element or automaton) possesses one asymmetric key pair, namely a *public* and a *private* key. The former may be made publicly available, provided that the latter remains private, but the knowledge about the public key does not allow derivation or computation of the private key. The Certification Authority (CA) [13,14], is the trusted authority responsible for creating and providing the corresponding PKC, which binds the specific public key with the identity of the entity. The private key is used to form digital signatures and is known only to the entity. Keys and certificates can be stored or transferred in hard disks, smart cards, diskettes, etc.

One of the main advantages of public key systems against symmetric key systems is the number of keys required for authentication and encryption. In the first case, the number of keys grows linearly with the number n of communicating network entities, while in the second case is in the order of n^2. Therefore, the costs of key generation and distribution, associated with the introduction of a new network

entity, are lower in the public cryptosystems and as a consequence, these systems are far more scalable and effective, especially as the number of participants is increasing rapidly. Additionally, asymmetric encryption can support non-repudiation services providing evidence that a specific action occurred, in contrast to symmetric encryption, where both the originator and recipient share the symmetric encryption key and consequently either party can generate the proof.

Attribute Certificates (ACs) [14] proposed by the Internet Engineering Task Force (IETF) PKI Working Group for carrying authorization information, are considered as a better alternative to X.509 Public Key Certificates (PKCs). ACs are issued and consequently signed by Attribute Authorities (AA). The characteristics of a distinct entity (person or automaton) specifying various attributes or properties, like group membership, role, security clearance, or other authorization information, are bind to that entity by digitally signing the appropriate AC. Thus, access to system resources, robust role-based authorization mechanisms, and access controls policies, can be effectively controlled by ACs [15]. It is worth noting that two well known general purpose authorization systems that use ACs are *Permis* [16] and *Akenti* [17], and that AC based authorization is also an extension to the IETF TLS protocol [23].

An access control mechanism decides whether a subject (e.g. user, automaton) is allowed certain rights on an object. Along with the Internet growth, requirements for access control have moved forward and old access control models were not any longer suitable for distributed, pervasive and mobile systems. To meet these challenges, distributed access control models and trust management emerged. Conventionally, in order for a system to make a decision about whether a user has the proper access privileges to the system's resources, the requester must, in the first place, submit a secret password to the system for verification. If the user passes the verification process, he is admitted into the system. Under these circumstances, most authorization decisions have left out in the hands of end systems (or intermediaries such as firewalls), which retain and implement Access Control Lists (ACLs) without the direct involvement of distributed security infrastructure components. However, as distributed authentication matures and gains global acceptance, it is an urgent need for systems to extend authentication-based technology to embody more sophisticated authorization features.

In this context, Role-Based Access Control (RBAC) [19] has received great attention as the most promising alternative to traditional discretionary (DAC) and mandatory (MAC) access controls. RBAC guarantees that only authorized or legitimate users, even if they are not previously engaged with a particular system, are given access to protected data or resources. On the other hand, attribute certification offers a straightforward and consistent method to extend identity-based public-key certification infrastructures to support role-based authorization policies. Towards this direction, it permits decentralized authorities to manage identities, role affiliations and permissions, as well as other authorization-related objects.

In a nutshell, some advantages of attribute certification in relation to other authorization mechanisms include:

(a) Precise representations of the role(s) authorized and currently active for a given user can be obtained, even for those users who were assigned multiple (or sometimes conflicting) roles. Users can designate their roles by enclosing or indicating the appropriate ACs when requesting a particular service. Of course, these bindings must be signed by the proper trusted authority.
(b) Role-based controls can be managed more flexible when attributes are represented by ACs thus separated from identity or public key certificates. This distinction corresponds appropriately to different management responsibilities, and allows authorization attributes to be generated and updated more frequently having an entirely temporary or per-transaction nature, when compared to the traditional authorization mechanisms which are considered static and rather inflexible.

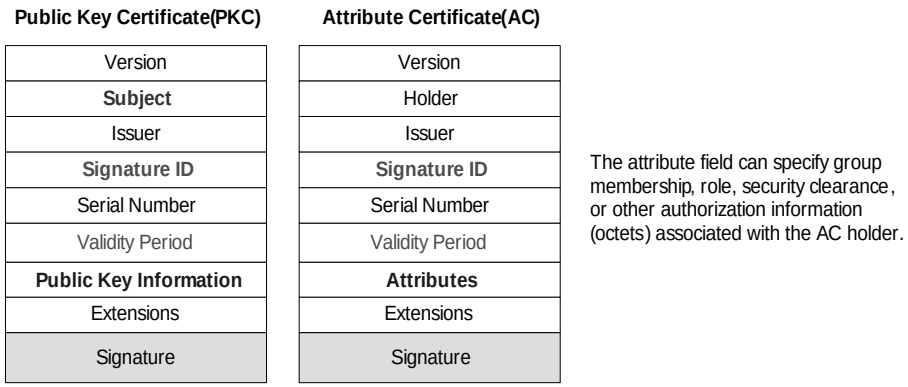

Fig. 1. Relation between a PKC and an AC and their basic structure.

(c) Enables the authorization objects to be represented in such a form that can easily be manipulated algorithmically. This is of particular value in cases when large numbers of attributes may apply to a given request. E.g., for rights obtained through inheritance or delegation.
(d) Allows a wide range of authorization decision criteria to be managed in a harmonized fashion. More specifically, it offers services that can be applied beneficially and efficiently to manage and delegate role-related attributes within distributed, mobile and mutually insecure computing environments, minimizing redundant trust in intermediaries.

More specifically, in the context of healthcare applications, ACs can effectively implement and support the RBAC schema, which mainly concerns the users' role (i.e. doctor, nurse, administration, management etc.) and the activities related to that. A role is a collection of application specific operations or procedures. The subjects, e.g. users or processes, derive their access rights and permissions from the role(s) they are assigned. Especially when dealing with sensitive medical data different access rights may be granted to the same type of users (i.e. a patient's attendant doctor may have additional access rights on the patient's record file in compare to another doctor of the hospital). The basic structure of an AC and its relation to a PKC are shown in Fig. 1. Of course, ACs acquisition procedure has also to be evaluated in terms of service times to further prove the feasibility of the proposed architecture/trust model. Consequently, experimental facts on this issue are provided in Section 5.

As mentioned earlier, ACs can also be used in the context of the *de-facto* TLS protocol, which is the predominant and most widely used security protocol in the wired Internet. After the user has obtained the corresponding AC he can employ it for the acquisition of the desired service in a totally protected TLS communication channel. This procedure can be exercised during or after the protocol's handshake phase, by simply pushing the correct AC to the appropriate application server (e.g. a Medical Picture Archiving and Communication Server PACS). However, this increased communication protection, enjoyed by the involved parties, comes at an extra cost in terms of service times and for that reason is thoroughly investigated in Section 6. This is particularly true especially when mobile devices are present and thus further study is essential to prove the feasibility of this proposal.

3. Healthcare trust model architecture

In this section, we present the architecture that is necessary to provide AC based transactions in a healthcare environment. There are four logical domains or subsystems in our model depicted in Fig. 2,

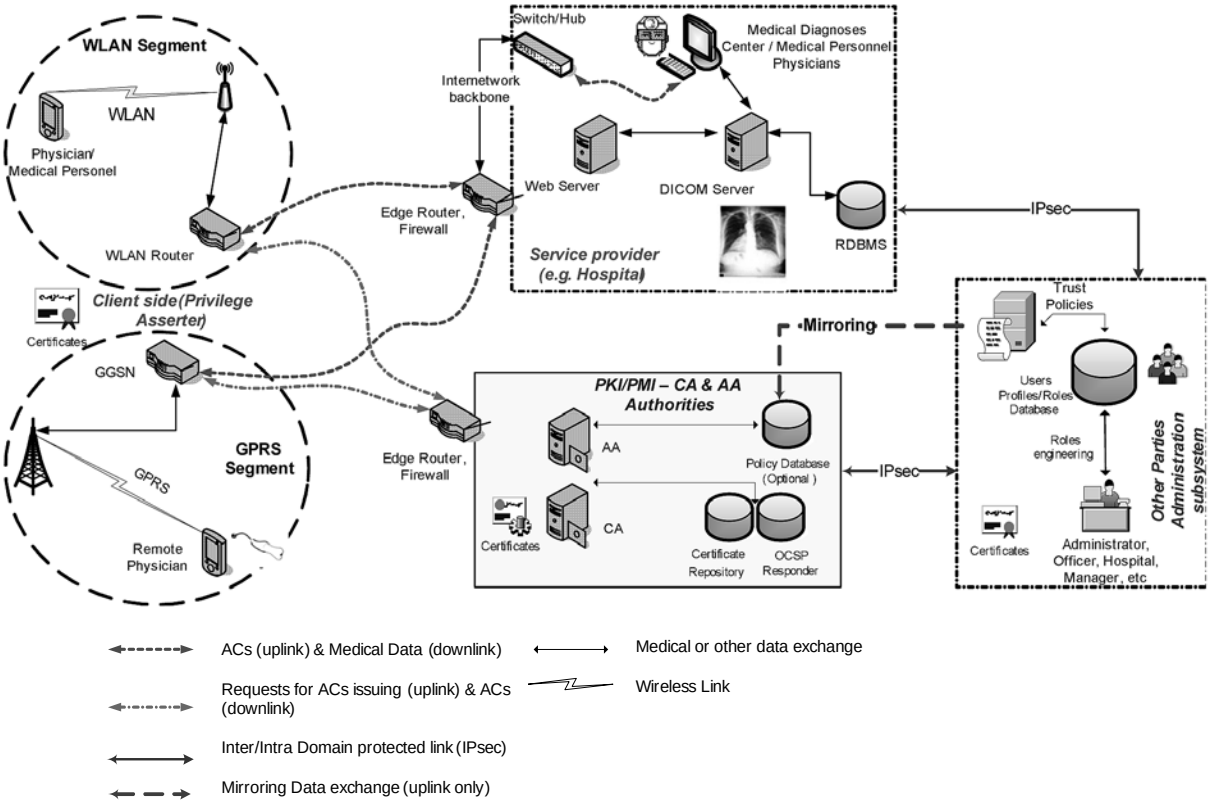

Fig. 2. Trust model for e-medical applications architecture.

namely the *client or privilege asserter*, the *server* (service provider), the *PKI* and the *administration* (other parties) subsystems. The first subsystem is a standalone domain, but the last three subsystems may belong to one corporate domain, or may be distributed in three or more interworking sub-networks. In particular, the main activities of these four logical domains are:

1. The client side is authenticated by its PKC and requests services bound by the appropriate ACs (credentials) that it holds.
2. The server side provides the services requested by the client. These services may include transfer of multimedia content, delivery of web content, file management, etc. A service can be provided or authorized if the client holds a valid and appropriate AC.
3. The PKI/PMI[1] subsystem issues and signs public key certificates and attribute certificates.
4. Hospital Information System (HIS) *administration* subsystem, which is responsible for the definition of the *trust policy* and administration of the whole system. More specifically, they are responsible for:

   (a) the definition of roles (e.g. attendant doctor, nurse, patient admission registrar's employee, clinic/hospital manager, etc.).

[1] AA entities are Privilege Management Infrastructure (PMI) elements. In this paper, we may use the term PKI not only for CA, but also for AA domains.

Table 1
Example of a role – permissions table (X means "Permission granted")

Role →	Attendant Doctor	Clinic Doctor	Hospital Doctor	Head of Clinic	Nurse	Head of Nurses	Patient's Reception	Hospital Manager	System admin	Other
Permissions	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001
Perm. 1									X	
Perm. 2							X		X	
Perm. 3				X				X	X	
Perm. 4		X	X	X		X	X	X	X	
Perm. 5		X	X	X		X	X	X	X	
Perm. 6	X	X	X	X		X	X	X	X	
Perm. 7	X	X		X		X	X	X	X	
Perm. 8	X	X	X	X		X	X	X	X	X
Perm. 9	X	X	X	X	X	X		X	X	X
Perm. 10	X	X	X	X	X	X	X	X	X	X

(b) the assignment of roles to persons or other entities.

The exact types and number of roles depend on the organizational structure of the provider's system and the level of its complexity. A role specifies what operations are allowed on the applications data (permissions) and the conditions for the occurrence of those operations. Furthermore, roles are bound with each service or application on the server side. Once a role is defined, an AC can bind that role with the person's identity or entity and associate the corresponding permissions with that person. Under this setting, it is possible to have two or more roles, corresponding to different ACs (in other words different or additional permissions), assigned to a person or entity. However, conflicting roles should be taken care of by the system [20,21]. For example, if the permissions of one role prevents access or modifications to a patient's medical record and another role allows it, the system must prohibit the same entity from being assigned those conflicting roles at the same time. Thus, it may be acceptable for a person to be a member of either an attendant doctor role, authorized to input data, or a plain doctor role, authorized to review data in different patients or clinics, but unacceptable to take on both roles within the same patient/clinic. In Table 1, we present a possible sketch for the role/permission table for a HIS with ten roles defined. Each role is represented by a bit pattern as depicted in the first row of the table. For instance, if a system user has been assigned 2 of the roles (e.g. Clinic Doctor and Head of Clinic), then the *attributes* field in the AC assigned to that user will have value (0001, 0011).

ACs are delivered to the users at the request of the service provider or the users. When the provider initiates AC issuing, he forwards the requests for particular ACs, corresponding to specific roles or even temporary actions, to the AA. AA validates the requests, creates the correct ACs, updates its Database (DB) and forwards them to the proper user(s). When necessary, a user can also request an AC on-the-fly from the serving AA. The request, as well as the issued AC, has to be compatible with the roles previously assigned to that user, so that conflicts are avoided. Finally, it worth noting that it is possible to issue ACs of limited duration. For example, providing service access to a visiting doctor for the duration of his visit or even temporary (transaction-oriented) access to system services are possible under this option. One of the advantages of these temporary certificates, having a short life, is that they do not usually need to be revoked and will therefore require not be included in any Certificate Revocation List (CRL).

In Fig. 3, we present three types of such potential authenticated requests and an AC sample. The fields indicated in the sample requests are constructed for demonstrative purposes, and it is up to the system designer to define the appropriate request structure that satisfies most of the requirements of the system at hand. Additionally, the applications could be adjusted to handle any specific type of request. In our scenarios, presented at a later section, we used three of those fields: the "RequestID" field, the

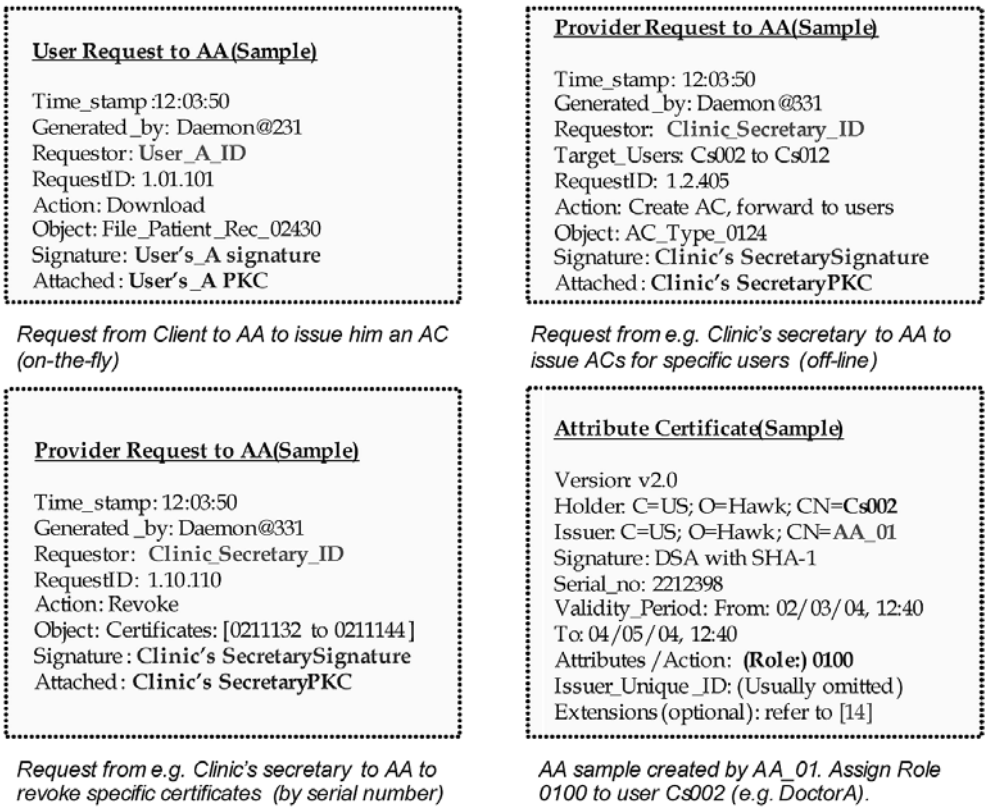

Fig. 3. Sample attribute certificate and requests.

"Action" field and the "Object" field. For instance, the "RequestID" field in the form of a bit-pattern could designate a unique combination of a role and permission in the roles/policy DB. The designer can appropriately tune this field, enabling a plethora of possible classifications. Furthermore, the "Action" field can specify different actions like "Revoke", "Download", "Read", "Update", "Approve", etc., in relation to the "RequestID" field. Finally, the "Object" field may define the resource or record that the "Action" field is applied to. For instance, a specific request might be "Download" cardiovascular related history record of the patient with a specific ID.

The procedure of AC delivery is well-secured since the requests are signed by the issuer's private key (Request || Digital_Signature)[2] and can be transmitted in clear-text, as they are actually useless to anyone who intercepts it. Naturally, to protect against replay attacks, the requester must include to the requests generated either a sequentially incremented field or a time stamp. Unfortunately, an attacker is still able to exploit Denial of Service (DoS) or Distributed DoS attacks (DDoS) attacks, e.g. by intercepting and tampering the requests or the issued ACs. For these kind of threats, firewall and intrusion detection systems are essential accordingly to ensure the availability of medical data. Other sort of attacks or threats, expressly for ACs, can be found in [14].

Several additional considerations regarding the architecture depicted in Fig. 2 are necessary. Ideally, PKI/PMI subsystems should not coexist in the same domain, since the authority that issues PKCs is

[2] $\text{Hash}_{(16\_\text{bytes})}$ = MD5(Request) and Digital_Signature = $(\text{Hash}_{16\_\text{bytes}})_{\text{Issuer's\_Private\_Key}}$.

usually quite different from the authority that issues ACs (PMI). However, in some implementations CA and AA functionality can be combined, as is the case of Fig. 2, where CA and AA are located in the same domain or machine. The administration and maintenance procedures of PKI subsystem can be provided by the Hospital MIS administration, or a Trusted Third Party (TTP), also known as the Certification Service Provider (CSP).

According to our model, for every sensitive transaction, an AC is requested, and this is performed possibly *on-the-fly*. Thus, it is essential to have the AA check against the roles/policy DB, so that the delivery of the correct AC is reassured. There are two possible ways to perform this check; by dynamically contacting the DB, or by incorporating an image of the DB in the PKI/PMI subsystem and offline updating this image, once, or several times during the time period of the day. In both cases, there is a strong requirement to secure the communication link between the domains of the AA and the DB. A security protocol like IP Secure (IPsec) [22] can be used to secure the link, thus constructing a Virtual Private Network (VPN). For example, IPsec Encapsulating Security Protocol (ESP) in tunnel mode can be applied to offer integrity, confidentiality services, and protection against traffic analysis between these network entities. IPsec uses the Internet Key Exchange (IKE) protocol for peer authentication, and IKE can be configured to use public key based authentication with certificates or static pre-shared secrets.

A final option to assure that the correct permissions are assigned to the requesting user is to allow AA initially issue only role ACs (Role Assignment ACs – RAAC) towards all users. When a client needs to acquire a specific service, he forwards the analogous RAAC to the service agent, and then the service agent, probably using an Access Control Policy Server, queries the other agent's domain DB to see – based on the client's RAAC and the permissions assigned to that role – if the user is entitled to access the specific medical information. This role-to-permissions or role specification query can also be performed either on-line or in a local policy DB that is maintained by the service Access Control Policy Server. In all above scenarios, it is implied that the DB is adequately protected inside each domain and all sensitive data are stored in encrypted form.

Two network entities can effectively authenticate each other by exchanging their public key certificates. In the context of public key technology enabled protocols, like TLS or Secure Sockets Layer Protocol (TLS/SSL) [24] and IPsec, this yields robust authentication and end-to-end protected communication (see Section 6).

4. Enabling mobile access to medical data

Handheld devices such as Personal Digital Assistants (PDAs) and Tablet PCs are more affordable today than ever before, and due to their convenience, have started to penetrate working environments where users are frequently moving (e.g. within a hospital/clinic or outdoors), while constantly wishing to receive access to services.

In a typical mobile e-health application scenario, the user is connected to a computer based patient record system and browses or updates medical data according to his role access privileges. Although the patient record system resides in a fixed location (i.e. a Hospital, or a Healthcare Data Center) the user (i.e. the patient's attendant doctor) may be at different locations. Such a location can be either the hospital premises or a treatment/care center established at a sports facilities center, an ambulance, a medical treatment centre on an island, or an urban area. Due to this mobility, several kinds of network architectures may be used. In the present research we have evaluated both the 802.11 (for locations with WLAN infrastructure such as a hospital) and General Packet Radio Service (GPRS) networks (for random locations).

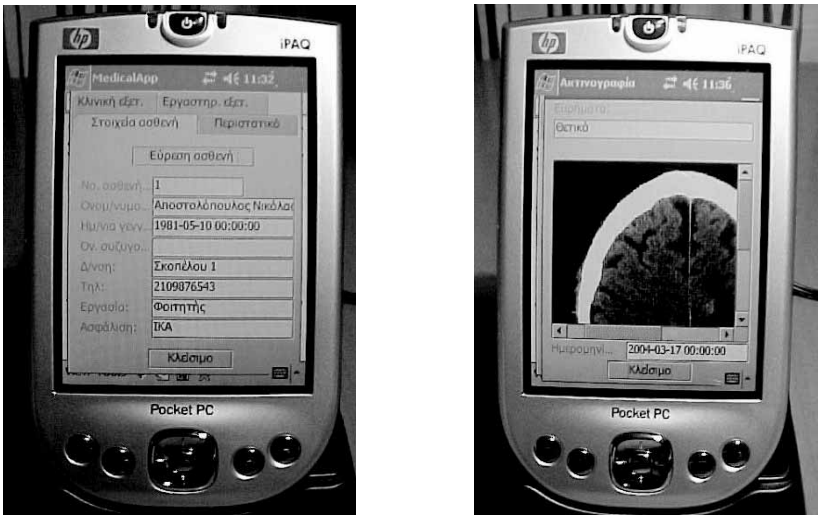

Fig. 4. Java application enabling access to electronic patient health record.

A Java application enabling access to a small incident based patient health record was developed for evaluating the proposed PKI mechanism [11]. The application runs in Compaq iPAQ H3970 Pocket PC allowing mobile access to patient's clinical data and DICOM compliant medical images (see Fig. 4).

Using the application the physician makes a selection, requesting data of a specific patient. A server agent (thread) is subsequently generated to dispatch the request. The server agent interactively asks the user to provide the AC corresponding to the requested service. Upon reception, the server agent has to validate the AC. First the AC's signature authenticity and origin is verified. It is implied that the certificate must be signed by an AA that the server agent trusts, while the public key certificates of all trusted CA/AAs can be kept in the server's cache memory. Next, the certificate's time expiration field is checked and finally, if appropriate, the server confirms that the AC is not included in the last retrieved Certificate Revocation List (CRL). Another option to test against revocation is the On-line Certificates Status Protocol (OCSP) responder. For example, the revocation of specific certificates may be requested by the hospital administration office. If the AC is valid and in accordance to the requested service, the server side provides the service. Otherwise, it can offer the following options to the doctor/user: (a) allow him to change his request, (b) allow the HIS administrator to adjust the user's role and provide him the appropriate AC at a later time, and finally (c) allow the user to request the required AC from an AA on-the-fly.

If the user selects the last option, the user agent (service/process/daemon) constructs on behalf of the user the appropriate request. After that, the user agent signs it with the user's private key and forwards it along with his public key certificate to the appropriate AA. AA shall validate the request (signature, expiration time, etc.), using the user's public key found in the user's public key certificate. Next, AA checks the user credentials by querying the provider's user policy DB (according to the provider's specific policies) or – in case of mirroring – its local roles/policy DB (see Section 3). So, in case the doctor has been assigned two or more roles, the main point of this process is to reduce the role set, so that the resulting group does not have any mutually exclusive permission. If everything is acceptable, AA shall issue and forward the corresponding AC back to the physician for immediate use.

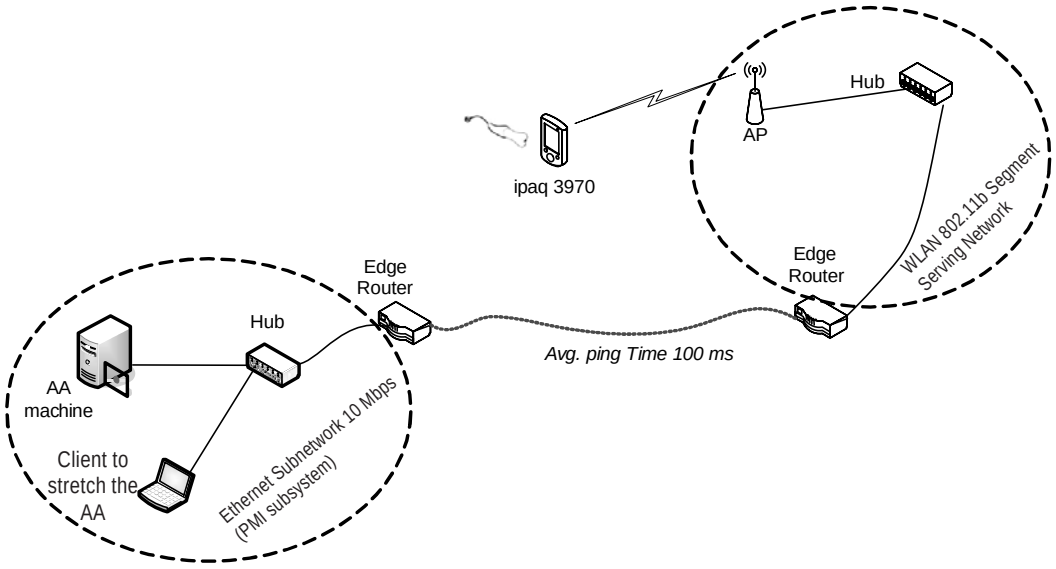

Fig. 5. Topology and test-bed with WLAN access.

5. Experiments on ACs delivering in mobile e-health scenarios

The scope of our research is to provide evidence that delivering ACs, and thus well-controlled access to healthcare applications and data, using mobile devices and networks is attainable with current technology. This will further support the feasibility of the discussed architecture and provide some experimental, indicative facts in terms of service times. Put another way, as mobile devices have relatively limited resources, like memory, storage and computational power, when compared to desktop machines, processor-demanding public key operations become a more questionable task when performed on such machines. Additionally, mobile networks still have very limited bandwidth when compared to their wired counterparts. Under these conditions, it is necessary to confirm the feasibility of our proposed model and architecture with current wireless networks and devices.

In this work, we used as a case study the delivery of ACs over GPRS and WLAN IEEE 802.11b access networks. The proposed architecture depicted in Fig. 2, was implemented with two experimental network architectures, illustrated in Figs 5 and 6. The only difference between these two topologies is the type of the access network the user is connected to. In Fig. 5, the access network is a WLAN, corresponding to a topology of a controlled healthcare environment, within the hospital premises, which are equipped with Wi-Fi network infrastructure. In Fig. 6, the user is connected via a commercial GPRS segment, covering a wide geographical area. In this case, remote access is enabled at random locations away from clinics or hospitals, e.g. at an accident site in case of an emergency. It is worth noting that the PMI subsystem may either be directly connected to the hospital's network backbone, functioning as a separate subsystem, or belongs to an external network domain, operated by a Certification Service Provider (CSP). Comparable test-beds for GPRS and WAP performance evaluation can be found in the literature [26–28].

In the second column of Table 2, we present the system characteristics of the devices used in our experiments, i.e. the handheld client, the AA server, the client that offers virtual load to the AA server, the WLAN access point, and the GPRS commercial network segment. Also in the third column of the same table we denote the software used for the development of the applications as well as their

Table 2
Characteristics of devices and applications

Device/Network	System characteristics	Software
Handheld Device Client	– Compaq iPAQ H3970 Pocket PC (PPC) – Windows PPC 2002 operating system – 400 MHz Intel X-Scale PXA250 CPU – 64 MB of RAM and 48 MB of flash ROM + a user-accessible section of 22 MB ROM for data, applications, and other files – Nokia D211 dual GPRS class 7/WLAN IEEE 802.11b PCMCIA card	– Microsoft's Embedded C++ version 4.0 – Open-source Apache-style license OpenSSL toolkit, version 0.9.7b (http://www.openssl.org) – 100 Kbytes of RAM
AA server	– Dual Pentium Xeon 2.4 GHz processors – 512 MB of RAM – Windows XP professional operating system	– Microsoft's Embedded C++ version 4.0 – Open-source Apache-style license OpenSSL toolkit, version 0.9.7b (http://www.openssl.org) – 96.1 Kbytes of RAM
Client for Virtual Load	– Celeron 1.2 GHz processor – 256 MB RAM	– Microsoft's Embedded C++ version 4.0 – Open-source Apache-style license OpenSSL toolkit, version 0.9.7b (http://www.openssl.org) – 100 Kbytes of RAM
Access Point	– D-link DWL-900AP+ – 11 Mbps	
GPRS Segment	– GPRS Coding Scheme 1 (9.05 Kbps) – Timeslots varying from 3 to 4 – Wireless network speeds from 27 to 36 Kbps	

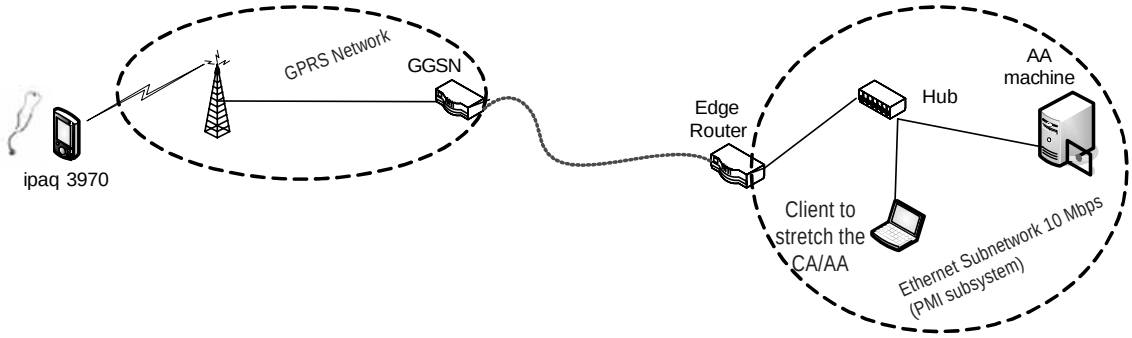

Fig. 6. Topology and test-bed with GPRS access.

requirements in RAM memory space. The well-known OpenSSL toolkit [29] was used to make our applications public key enabled and create the necessary certificates for our experiments.

The AA server process is multi-threaded, creating a new thread to serve each incoming AC request. The total size of the client's request is about 733 bytes. Beside the client process, running on the handheld device client, we also used another multi-threaded process located at another client to load the AA with virtual requests for AC certificate issuing. This client device is another laptop machine with the characteristics shown in Table 2, and is wired to the PMI sub-network with a speed of up to 10 Mbps. We experimented with various virtual load values offered to the AA, ranging from 20 to 60 requests per minute, but and the effect on the performance was negligible. Measurements were gathered from a set of 2000 transactions between the CA/AA server and the handheld client. Our experiments were conducted in different days and hours during a week period while 50% of the measurements were logged during peak hours both in WLAN and GPRS networks.

Table 3
Description of the service times measured

Client Side	
Request Creation Time (RCT)	Time for the user's device to create the request. This is done by a daemon/process/service installed in user's device.
Request Overall Time (ROT)	Elapsed Time from request transmission until the AC has been received.
AA Side	
Request Verification Time (RVT)	Time for the AA to verify the request (recalculate hash, validate signature)
Attribute Authority certificate Creation Time (ACT)	Time for AA to create the AC, according to the request.

Table 4
Average service times in milliseconds for scenarios I & II

	Time	RCT	RVT	ACT	ROT
WLAN	Average	1094.5	10.6	68.5	3024.3
	St. Deviation	31.9	16.2	11.4	198.3
GPRS	Average	1109.6	11.5	62.7	4382.9
	St. Deviation	32.6	6.2	8.3	1897.6

We tracked and measured the times shown in Table 3 at the client and server applications. At this point, it is important to note that the average ping times between the subnetwork of the PMI subsystem and the serving access subnetwork is 100 milliseconds for the WLAN architecture of Fig. 5 and 1230 milliseconds for the GPRS architecture of Fig. 6. This difference has a significant, but inevitable, impact on the user perception about the service quality offered by those distinct implementations. Additionally, in the GPRS implementation, there is an extra network delay derived from the fact that the AA server is not located at the service provider's core network. As a result, the request, as well as the AC, has to traverse all the way back to the local network where the AA is located. Considering the average ping time between those domains we can presume that the extra time needed for the completion of the request is at least 2 round trip times higher.

The average values and standard deviations of the time durations measured are presented in Table 4. As we see, the average client's total time for one transaction (one AC issuing), is about 3.0 sec and 4.4 sec, when the user is using a WLAN network or he is connected via GPRS, respectively. The two main comments on these results are:

1. The total time (ROT) to service one client request mainly depends on the time needed for creating the request at the client and the network transfer times. The serving times at the server side are quite insignificant. Naturally, ROT is expected to grow depending on the sub-networks distance expressed in ping times. The more the numbers of domains the request has to travel, the higher the ROT is expected to be.
2. The increased standard deviation times of the total time ROT at the GPRS implementation is mainly due to the instability of the GPRS connection. However, the network speeds for 3G mobile communication networks will be 144 Kb/s up to 348 Kb/s for wide and up to 2 Mb/s for low coverage and mobility, which will substantially reduce transfer times. On the other hand, the corresponding WLAN standard deviation time is typical for this type of connection.

Finally, we can conclude that in both implementations the total time to service AC requests is quite low and comparable to other services offered to wireless and mobile users.

6. Experiments on transferring medical data with TLS protocol

As already mentioned in Section 2, AC based authorization is an extension to the TLS protocol, as ACs can be used either during or sometime after the initial TLS protocol handshake. Although, TLS is the predominant protocol in the wired Internet, and thus it can be easily employed in the health sector, mostly, the handshake phase is performed from one direction only; the client authenticates the server. Consequently, it is considered necessary to demonstrate to what cost comes this "blending" of the TLS protocol with ACs when performing a full handshake. In this section, we present service time measurements that prove both the feasibility of the demanding two-way handshake phase and the ability of wireless (GPRS, WLAN) links to transfer various sizes of DICOM compliant image data over TLS protected links.

Once again, we used the two test-beds depicted previously in Figs 5 and 6. User, server and network devices remained unaltered. We tracked and logged two distinct service times concerning (a) the total TLS handshake time and (b) the overall transfer time of DICOM images. As previously mentioned, the measurements were performed during different days and peak hours times. Average times for each different scenario were calculated based on 500 samples sets in WLAN case and 20 samples sets in GPRS. In both scenarios, the AA server was stretched with virtual requests for TLS handshakes and TLS protected transfers, ranging from 1 to 2000 requests per second in the WLAN case and from 1 to 20 requests per second in the GPRS case.

Similarly, the client and server applications in Embedded C++ Version 4 and employed the OpenSSL toolkit in version 0.9.7b to make them SSL enabled. Additionally, we tried to minimize the client's application demand in processing power and memory capacity, by removing dispensable memory and power consuming calls to OpenSSL functions. Therefore, we left out functions that load libraries with error strings, verify certificates paths, certificates chain depths above two and we enabled the client to support only SSLv3 and TLSv1. Finally, we excluded from the client and the server, support for ciphers and Message Authentication Codes (MACs) algorithms that are generally considered anonymous or weak, e.g. MD5.

All RSA keys are 1024 bits in length, pre-master secret exchange is based on ephemeral Diffie-Hellman key with RSA signatures, thus supporting forward secrecy, and the resulting symmetric TLS session key is 256 bits long. The complete cipher suite algorithms that our applications employed are: EDH-RSA for key exchange with key size 512 or 1024 bits, AES256 for encryption and HMAC-SHA-1 for integrity checksums (DHE-RSA-AES256-SHA). This suite can be characterized as heavy, compared e.g. to weaker but faster Kilobyte-SLL's cipher-suites options (RSA_RC4_128_MD5 and RSA_RC4_40_MD5).

The results for the WLAN 802.11b case are presented in Fig. 7, while the corresponding GPRS average times durations are presented in Fig. 8. As shown in these figures, the effect on the overall total handshake time is almost insignificant. In the WLAN case, this time seems to fluctuate around 1.175 seconds, at virtual loads higher than 10 requests per second. The main comment on the results of Figs 7 and 8 is that although current WLAN technology can support such highly secured information transfer, offered by TLS protocol, there is still a long way until mobile communications can support the same service. These high values were somewhat expected due to the unreliable nature and the low transfer speed of the GPRS service. Several other works conducted with similar mobile devices and comparable network setups and procedures verify this observation [32–35]. However, as mentioned earlier, the expected network speeds for 3G-and-beyond mobile systems will substantially reduce transfer times and make this advanced secure transfer possible. Other optimizations, excluding hardware evolution, such as compression, buffering techniques [31], and other solutions proposed in [35], can bring TLS-based data transfer closer to reality in mobile environments.

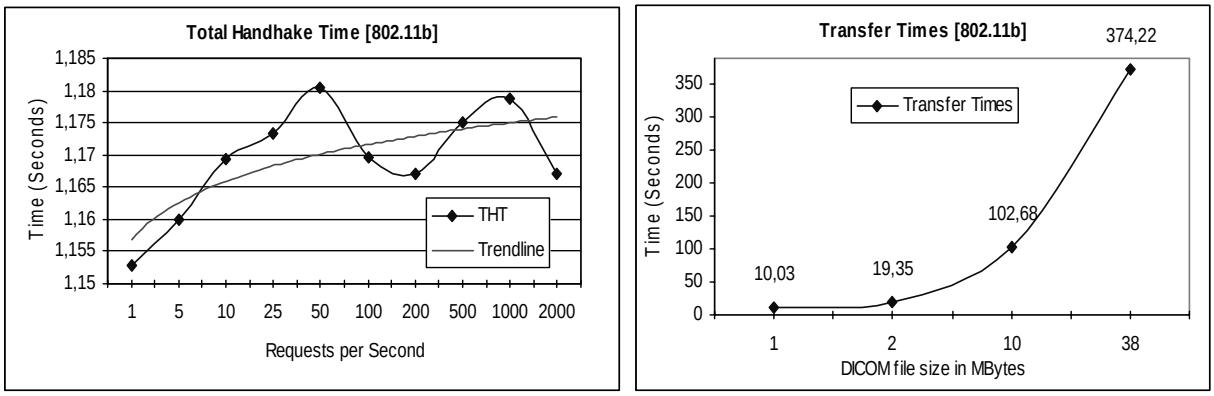

Fig. 7. WLAN average time durations (handshake and transfer times).

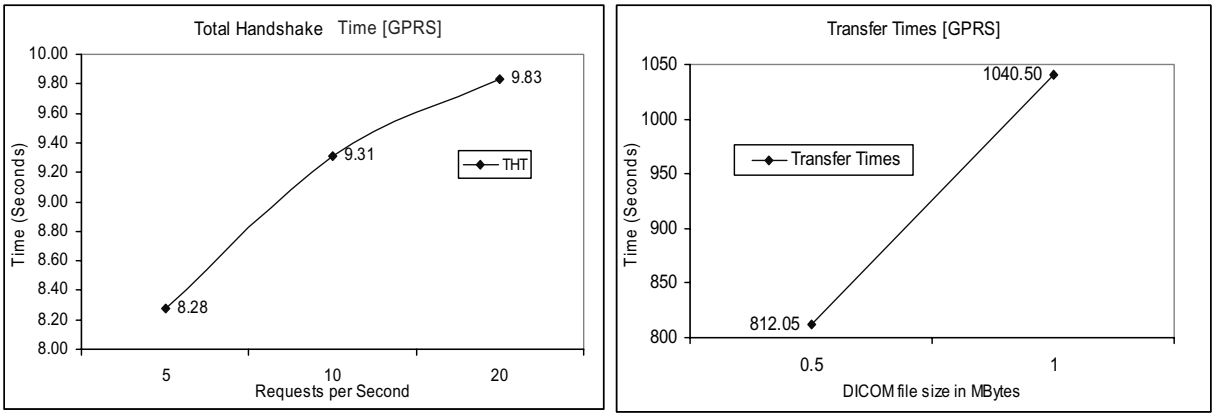

Fig. 8. GPRS average time durations (Handshake and Transfer times).

As a final point, battery or energy consumption during TLS protected sessions needs also consideration. This is even more important due to the mobile devices' battery limitations. Several works, mentioned in [35], can be found in the literature that cover to a great deal this issue and show that TLS protected sessions can be quite affordable for present and future handheld devices.

7. Conclusions and future work

The increasing commercial use of laptops, Tablet PCs and PDAs makes possible the mobile paradigm to continuously ongoing users in the health sector. Mobile systems assist health professionals in accessing patient medical information fast, while they move freely in their work environment. From the developers approach, the increase in mobile networks speed and PDAs storing and processing power and the ability to set up wireless networks in hospitals and clinics assists the development of mobile healthcare applications.

Moreover, with the advent of 3G-and-beyond systems like Universal Mobile Telecommunications System (UMTS), and the recently emerging Wi-MAX technology, mobile medical services can fully exploit and take advantage of robust PKI-oriented services that until now only wired segments have the

chance to employ. However, the security aspect of such applications is considered a critical issue due to the sensitive and private nature of the medical information.

In this paper we provide evidence that delivering ACs, and thus well-controlled access to healthcare applications and data, using mobile devices and networks is attainable even with current technology. This is supported both for ACs acquisition procedures, as well as in using mutually authenticated and protected TLS sessions along with ACs to transfer sensitive medical data. On the other hand, the proposed model has to be further evaluated in terms of its security robustness and strength. This task includes the identification, detection and remediation of the possible threats that the anticipated model may face. Another issue, currently under inquiring, is TLS protocol comparison, in terms of performance, with other analogous public key enabled security protocols, like IPsec and Secure Shell (SSH), to transfer medical data when employed in wireless realms.

References

[1] R. Andrade, Aldo von Wangenheim and M.K. Bortoluzzi, Wireless and PDA: a novel strategy to access DICOM-compliant medical data on mobile devices, *International Journal of Medical Informatics* **71** (2003), 157–163.

[2] J. Reponen et al., Initial experience with a wireless personal digital assistant as a teleradiology terminal for reporting emergency computerized tomography scans, *J. Telemed. Telecare* **6**(1) (2000).

[3] K. Hung and Y. Zhang, Implementation of a WAP-Based Telemedicine System for Patient Monitoring, *IEEE Transactions on Information Technology in Biomedicine* **7**(2) (2003), 101–107.

[4] L.G. Yamamoto and L.K. Shirai, Instant telemedicine ECG consultation with cardiologists using pocket wireless computers, *Amer. J. Emerg. Med.* **19**(3) (May 2001), 248–249.

[5] P. Giovas et al., Transmission of electrocardiograms from a moving ambulance, *J. Telemed. Telecare* **4** (1998), 5–7.

[6] K. Shimizu, Telemedicine by mobile communication, *IEEE Eng. Med. Biol. Mag.* **18**(4) (1999), 32–44.

[7] R.H. Istepanian et al., Design of mobile telemedicine systems using GSM and IS-54 cellular telephone standards, *J. Telemed. Telecure* **4** (1998), 80–82.

[8] S. Khoor, K. Nieberl, K. Fugedi and E. Kail, Telemedicine ECG-telemetry with Bluetooth technology, *Computers in Cardiology* (2001), 585–588.

[9] E. Hall et al., Enabling remote access to personal electronic medical records, *IEEE Engineering in Medicine and Biology Magazine* **22**(3) (2003), 133–139.

[10] C. Finch, Mobile computing in healthcare, *Health Management Technology* **20**(3) (April 1999), 63–64.

[11] I. Maglogiannis, N. Apostolopoulos and P. Tsoukias, Designing and Implementing an Electronic Health Record for Personal Digital Assistants (PDA's), *International Journal for Quality of Life Research* **2**(1) (2004), 63–67.

[12] A. Nash, W. Duane, C. Joseph and D. Brink, *PKI Implementing and Managing E-Security*, Berkeley: RSA press, 2001.

[13] PKIX Working Group, Public-Key Infrastructure (X.509) (pkix), Last Modified: 2004-09-07, electronically available at: http://www.ietf.org/html. charters/pkix-charter.html.

[14] S. Farrell and R. Housley, An Internet Attribute Certificate Profile for Authorization, IETF RFC 3281, 2002.

[15] R. Oppliger, G. Pernul and C. Strauss, *Using Attribute Certificates to Implement Role Based Authorization and Access Control Models*, in the Proc. of 4. Fachtagung Sicherheit in Informationsystemen (SIS 2000), Zurich, Switzerland, 2000, 169–184.

[16] D. Chadwick, The PERMIS X.509 Based Privilege Management Infrastructure, IETF Internet Draft, <draft-irtf-aaaarch-permis-00.txt>, April 2002.

[17] M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson and A. Essiari, *Certificate-based Access Control for Widely Distributed Resources*, in Proceedings of the 8th USENIX Security Symposium, Washington, DC, 1999.

[18] A. Arseanult and S. Turner, Internet X.509 Public Key Infrastructure: Roadmap, PKIX Working Group, IETF Internet Draft, <draft-ietf-pkix-roadmap-09.txt>, July 2002.

[19] D.F. Ferraiolo, J.A. Cugini and R.D. Kuhn, Role-Based Access Control (RBAC): Features and Motivations, available at: http://hissa.ncsl.nist.gov/rbac/newpaper/rbac.html, 1995.

[20] D.F. Ferraiolo, R. Sandhu, E. Gavrila, D.R. Kuhn and R. Chandramouli, Proposed NIST Standard for Role-Based Access Control, *ACM Transactions on Information and System Security* **4**(3) (2001), 224–274.

[21] R.S. Sandhu, E.J. Coyne, H.L. Feinstein and C.E. Youman, Role-Based Access Control Models, *IEEE Computer* **29**(2) (1996), 38–47, IEEE Press, electronically available at: http://csrc.nist.gov/rbac/sandhu96.pdf.

[22] S. Kent and R. Atkinson, Security Architecture for the Internet Protocol, RFC 2401, Nov. 1998.

[23] T. Dierks and C. Allen, The TLS Protocol Version 1.0, IETF RFC 2246, Jan. 1999.

[24] A. Frier, P. Karlton and P. Kocher, The SSL 3.0 Protocol Version 3.0; Available at http://home.netscape.com/eng/ssl3/draft302.txt.
[25] M. Myers et al., X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP, IETF RFC 2560, 1999.
[26] R. Chakravorty, J. Cartwright and I. Pratt, *Practical Experience with TCP over GPRS*, in the Proc. of IEEE GLOBECOM 2002, Taipei, Nov. 2002.
[27] J. Korhonen, O. Aalto, A. Gurtov and H. Laamanen, *Measured Performance of GSM HSCSD and GPRS*, in the Proc. of the IEEE Int'l Conf. On Communications (ICC'01), Helsinki, June 2001.
[28] G. Kambourakis, A. Rouskas and S. Gritzalis, Performance Evaluation of Public Key Based Authentication in Future Mobile Communication Systems, *EURASIP Journal on Wireless Communications* (*JWCN*) **1**(1), 184–197.
[29] J. Viega, M. Messier and P. Chandra, *Network Security with OpenSSL*, O'Reilly, 2002.
[30] S. Farrell, TLS extensions for attribute certificate based authorization, TLS Working Group, Internet Draft, <draft-ietf-tls-attr-cert-00.txt>, Feb. 1998.
[31] E. Rescorla, *SSL and TLS Designing and Building Secure Systems*, Addison-Wesley, 2001.
[32] V. Gupta and S. Gupta, *Experiments in Wireless Internet Security*, in the Proc. of IEEE Wireless Communications and Networking Conf. (WCNC 2002), no. 1, March 2002, 859–863.
[33] J. Al-Muchtadi, D. Mickunas and R. Campbell, A Lightweight Reconfigurable Security Mechanism for 3G/4G Mobile Devices, *IEEE Wireless Communications* **9**(2) (April 2002).
[34] A. Harbitter and D. Menasce, *The Performance of Public Key-enabled Kerberos Authentication in Mobile Computing Applications*, in the Proc. of ACM Conf. on Computer and Communications Security (CCS), Pennsylvania, Nov. 2001, 78–85.
[35] G. Kambourakis, A. Rouskas and S. Gritzalis, Experimental Analysis of an SSL-based AKA mechanism in 3G-and-beyond Wireless Networks, *Kluwer Wireless Personal Communications* (*WPC*), *special issue on Security for Next Generation Communications* **29**(3–4) (June 2004), 303–321, Kluwer Academic Publishers.