Privacy-Preserving Electronic Health Records

Liesje Demuynck* and Bart De Decker

Department of Computer Science, K.U.Leuven
Celestijnenlaan 200A, B-3001 Leuven, Belgium,
{Liesje.Demuynck,Bart.DeDecker}@cs.kuleuven.be,
http://www.cs.kuleuven.ac.be

Abstract. Electronic health records enable the global availability of medical data. This has numerous benefits for the quality of offered services. However, privacy concerns may arise as now both the patient's medical history as well as the doctor's activities can be tracked. In this paper, we propose an electronic health record system which allows the patient to control who has access to her health records. Furthermore, provided she does not misuse the system, a doctor will remain anonymous with respect to any central authority.

1 Introduction

In e-health, new information and communication technologies are used to improve the quality of healthcare services while at the same time reducing the corresponding costs. This is, for example, achieved by electronic health records (EHRs), which allow for global availability of medical information in a standardized format. EHRs enable efficient communication of medical information, and thus reduce costs and administrative overhead. Furthermore, medical errors can be reduced significantly. In current healthcare systems, medical data can be interpreted in ambiguous ways. Moreover, a patient's health records can be dispersed over multiple sites without the healthcare professional having access to (or even knowledge of) this data. EHRs provide a solution to these problems.

There are, however, serious privacy concerns associated with the move towards electronic health records. Medical data should not only be protected against outsiders, but also against insiders. Studies have shown that patients do not trust central authorities with their medical data. They want to decide themselves who is entrusted with this data and who is not. These concerns are justified, as unauthorized secondary use of medical information, for example by an employer or for advertising purposes, can easily be achieved.

Next to patients, healthcare providers want their privacy to be protected. A central repository of medical data controlled by strong access regulations allows for the monitoring of a doctor's actions. Central authorities can track down who is treated by which doctor, how, and for what reasons. Hence, patient-doctor autonomy is disrupted.

* Research Assistant of the Research Foundation - Flanders (FWO - Vlaanderen)

Unfortunately, current technologies abstract away from privacy concerns in order to obtain both secure and efficient health record systems. In this paper, we propose a system which is both secure and privacy-preserving. The system protects the patient's privacy by allowing her to control who has access to her medical information. However, no personal information can be hidden from the doctor entrusted with this access. Furthermore, a doctor's privacy is conditionally preserved: unless abuse is detected, no central authority knows which patient is treated by which doctor and for what purposes.

The remainder of this paper is structured as follows. First, the building blocks used in the system are introduced. Afterwards, we describe the system itself and evaluate its properties. Finally, we conclude the paper with a brief discussion of related work and a summary containing the major conclusions and future work.

2 Basic Building Blocks

2.1 Cryptographic Hash Functions

A good hash function $\mathcal{H}$ resembles a random function as much as possible. It takes an input of arbitrary length and maps it to an output of fixed length. Hash functions are efficiently computable but hard to invert. Also, it is difficult to find two inputs mapping onto the same output.

2.2 The RSA Function

The RSA function [11] for an instance (n, v) is a trapdoor one-way permutation in $\mathbb{Z}_n^*$ defined as $\mathrm{RSA}_{(n,v)} : w \mapsto w^v \mod n$. Here, value n is constructed as the product of two random primes p and q with binary length $|p| = |q| = |n|/2$. Value v is randomly chosen and relative prime to $\phi(n) = (p-1)(q-1)$.

The function is efficiently computable and easy to invert if $v^{-1} \mod \phi(n)$ is known. It is assumed that in all other cases, the RSA function is hard to invert.

In the remainder of this paper, we will denote the execution of i subsequent applications of $\mathrm{RSA}_{(n,v)}$ to an initial value w as $\mathrm{RSA}^i_{(n,v)}(w)$, with $\mathrm{RSA}^0_{(n,v)}(w) = w$. Note that $\mathrm{RSA}^i_{(n,v)}(w) = \mathrm{RSA}_{(n,v^i)}(w)$ for each $i \in \mathbb{N}$.

2.3 The Guillou-Quisquater Proof of Knowledge

A Guillou-Quisquater proof of knowledge [7] is an interactive protocol between a prover P and a verifier V. The inputs to the protocol are public values x and (n, v). After successful execution, V is convinced that P knows a value w such that $w = \mathrm{RSA}^{-1}_{(n,v)}(x)$. In addition, the only thing V can learn from this protocol execution is whether or not P knows such a w.

In the remainder of this paper, we will denote the Guillou-Quisquater proof for an instance (n, v^i) with $i \in \mathbb{N} \setminus \{0\}$ as $\mathrm{GQProof}\{\mathrm{RSA}^{-i}_{(n,v)}(x)\}$.

2.4 Verifiable Encryption

A verifiable encryption scheme [13, 1, 3] is an interactive two-party protocol between a prover P and a verifier V. The public input of the protocol is a public encryption key pk and a value x with $(w, x) \in \mathcal{R}$ for a one-way relation $\mathcal{R}$ and a secret value w only known to P. After successful execution, V obtains an encryption of w under public key pk.

A verifiable encryption ensures the verifier that the encrypted value w is as such that $(w, x) \in \mathcal{R}$ for the specified relation $\mathcal{R}$ and public value x. As a consequence, it also convinces V that the prover knows a secret value w corresponding to x. Moreover, the protocol does not reveal any additional information about w to V than what she already knew beforehand. In particular, if V does not know the private key sk corresponding to pk, then she cannot find out w.

A verifiable encryption protocol can be created for the RSA relation $(w, x = \mathrm{RSA}^{i}_{(n,v)}(w))$. In the remainder of the paper, this encryption will be denoted as $\mathrm{VEncrypt}_{pk}\{w, w = \mathrm{RSA}^{-i}_{(n,v)}(x)\}$.

2.5 Anonymous Credential Systems

Anonymous credentials [4, 2] allow for anonymous yet accountable transactions between users and organizations. Here, a simplified version of the system is presented. In particular, not all functionality is described and abstraction is made of the use of pseudonyms. Also, note that anonymous credential systems should be built on top of anonymous communication channels [5, 10].

Credential Issuing. An organization can issue a credential to a user. This credential may contain attributes such as a name, address or expiration date. After successful execution of the issue protocol, the user receives a non-transferable credential *Cred* and the organization receives an issue transcript. The issue protocol will be denoted as $\mathrm{getCred}(attrlist) \rightarrow Cred; GetTrans$.

Credential Showing. The user proves to an organization that she is in possession of a credential *Cred*. In addition, she selectively discloses some attributes to the verifier. The result of the protocol is a transcript *ShowTrans* for the verifier. Different transcripts (and thus different shows) of the same credential cannot be linked to each other or to their corresponding *GetTrans*. During a show protocol, the user may decide to enable some additional options; she may sign a message *Msg* with her credential, which provides a provable link between *ShowTrans* and this message. In addition, she might enable *ShowTrans* to be deanonymizable. Upon fulfillment of the condition *DeanCond*, this allows for a trusted deanonymizer to recover the corresponding transcript *GetTrans*, which might then be used to identify the user. In the sequel, the show protocol will be denoted as $\mathrm{showCred}(Cred, [attrs], [DeanCond)], [Msg]) \rightarrow ShowTrans$.

Credential Revocation. A credential can be revoked by its issuer. This is denoted as revokeCred($GetTrans$).

3 Description of the System

We first give an overview of the system's requirements, roles and protocols. Afterwards, the construction of these protocols is described in detail.

3.1 Requirements, Roles and Protocols

Requirements. The system consists of anonymized electronic health records, which are stored in a central database. Each record contains medical information about a patient, signed by an approved but unknown healthcare professional.

To protect the patient's privacy, only authorized doctors may access the database. These doctors can read and inspect all the health records. However, unless they have gained the patient's trust, doctors should not be able to link different records of a patient to each other or to the patient. This trust must be complete, i.e. the patient must not be able to hide partial medical information towards a trusted doctor.

Doctors must enjoy full anonymity with respect to the system. It must not be possible for any central authority to track down which patient is treated by which doctor and for what purposes. However, when abuse of anonymity is detected, this anonymity should be revoked and appropriate actions should be taken. Types of abuse are, for example, illegal requests for a patient's health records or the submitting of incorrect health records.

Roles. An individual using the system is either a *doctor D* or a *patient P*. A doctor is assumed to live up to a deontological code and does not share any medical information about a patient with another doctor, unless both are entrusted with the care of this person. A special type of doctor is an *emergency doctor ED*. An emergency doctor works at an emergency room (ER) and hence needs special privileges.

The system itself consists of a *registrar R*, a *database manager DBMan*, and an *emergency service ES*. Next to this, a number of *deanonymizers* may be present. Deanonymizers judge and perform deanonymizations when abuse is detected. The registrar stores bookkeeping information and registers both patients and doctors. The database manager guides the retrieval and addition of health records from and to the database by performing the necessary access controls. Finally, the emergency service performs emergency retrieval of health records when the patient is unconscious and her doctor is unreachable.

Protocols. A patient entering the system first performs a *patientRegistration* with R. As a result, she obtains a list of private keys $sk_p(i)$ ($i \in \{1, \ldots, t\}$), which will be used at successive moments in time. P can now entrust a doctor D

with her medical information by executing a *visitDoctor* protocol with D. From then on this doctor will be able to manage all of her health records. If P wants to end this trust relation, she enables a new private key $sk_p(i+1)$ by performing the *changePrivateKey* protocol with R. As a consequence, D will no longer be able to add or retrieve any *new* health records concerning P.

A doctor registers with the system by executing the *doctorRegistration* protocol with R. This provides her with an access credential to the record database. Once entrusted by a patient, D can manage her health records by means of the *addHealthRecord* and *retrieveHealthRecords* protocols. Finally, a doctor working at ER may perform an emergency retrieval of a patient's medical data by using the *emergencyRetrieval* protocol.

3.2 Practical Construction

System Setup. A trusted third party TTP generates a strong hash function $\mathcal{H}$ and system parameters (n, v) for the RSA-function. Furthermore, the emergency service ES generates an encryption keypair (pk_{es}, sk_{es}). Private key sk_{es} is kept secret by ES, while values (n, v), $\mathcal{H}$ and pk_{es} are made public to all participants. As no participant may invert the RSA-function, TTP must make sure the factors p and q of $n = pq$ are discarded immediately after parameter generation.

Health Records. A health record is a show transcript *ShowTrans* generated as a result of a deanonymizable credential show by a doctor. The content of the record is a message of the form (*ID,medical data*), signed during the show protocol. *Medical data* is a text string representing the health information and *ID* is a unique identification tag created as $ID = \mathcal{H}(sk_p(i) \parallel j)$ for a counter value j and a patient's temporal private key $sk_p(i)$. To ensure the uniqueness of *ID*, each counter value is used only once for a temporal private key.

PatientRegistration. A patient entering the system first generates her (private keys, public key) pair $((sk_p(1), \ldots, sk_p(t)), pk_p)$. This is done in a preprocessing stage. Patient P chooses a suitable t and random value $x \in_{\mathcal{R}} \mathbb{Z}_n^*$. She then sets $sk_p(i) = \mathrm{RSA}_{(n,v)}^{t-i}(x)$ for $i \in \{1, \ldots, t\}$ and $pk_p = \mathrm{RSA}_{(n,v)}^{t}(x)$. Each of the private keys will be used at successive moments in time. Note that, given private key $sk_p(i)$, all previous keys $sk_p(j)$ with $j \in \{1, \ldots, i-1\}$ can be computed, but none of the future keys $sk_p(k)$ with $k \in \{i+1, \ldots, t\}$.

P then starts the registration procedure with R. In a first step, she identifies herself to R and provides her with a verifiable encryption $\omega_{sk_p(1)}$ of $sk_p(1)$, encrypted under the emergency service's public key. Note that $\omega_{sk_p(1)}$ implicitly proves her knowledge of $sk_p(1)$. Afterwards, P retrieves a credential binding her identity with her current private key $sk_p(1)$.

The registrar additionally stores some bookkeeping information for later use. In particular, she stores a specification $i = 1$ of the current private key $sk_p(i)$, a verifiable encryption of $sk_p(1)$ and the current value n_1 for the counter used when creating a new record *ID*. She also stores the credential's issue transcript.

1. P : $((sk_p(1), \ldots, sk_p(t)), pk_p) = \text{generatekeys}(t)$
2. $P \leftrightarrow R$: verification of P's identity
3. $P \rightarrow R$: $\text{send}(pk_p)$
4. $P \leftrightarrow R$: $\omega_{sk_p(1)} = \text{VEncrypt}_{pk_{es}}\{sk_p(1), sk_p(1) = RSA^{-1}_{(n,v)}(pk_p)\}$
5. $P \leftrightarrow R$: $\text{getCred}(\{\text{'patient'}, P, pk_p, 1\}) \rightarrow \mathit{Cred}_{sk_p(1)}; \mathit{GetTrans}_{sk_p(1)}$
6. R : $\text{store}(P, pk_p, \{1, \omega_{sk_p(1)}, \mathit{GetTrans}_{sk_p(1)}\}, \{1, n_1 = 0\})$

DoctorRegistration. A doctor registering with the system provides her identity and relevant university diplomas to R. The registrar checks this information, and, if approved, issues a doctor credential. This credential contains the doctor's specialties, such as, for example, the fact that she is an emergency physician.

1. $D \leftrightarrow R$: verification of identity and diplomas
2. $D \leftrightarrow R$: $\text{getCred}(\{\text{'doctor'}, \text{specialties}\}) \rightarrow \mathit{Cred}_d; \mathit{GetTrans}_d$
3. R : $\text{store}(\mathit{GetTrans}_d)$

VisitDoctor. Although a patient can visit her doctor anonymously, she must allow D to access her complete list of health records. Therefore, she gives D her private key $sk_p(i)$ and additionally proves that this is her current private key by showing both her credential and her public key pk_p. The resulting show transcript is then stored by D as a proof of the patient's trust. The private key $sk_p(i)$ can now be used by D to access the patient's health records.

1. $P \rightarrow D$: $\text{send}(i, sk_p(i), pk_p)$
2. $P \leftrightarrow D$: $\text{showCred}(\mathit{Cred}_{sk_p(1)}, \{\text{'patient'}, pk_p, i\}, \mathit{null}, \mathit{null}) \rightarrow \mathit{ShowTrans}$
3. D : $\text{check}(\text{RSA}^{i}_{(n,v)}(sk_p(i)) = pk_p)$
4. D : $\text{store}(\mathit{ShowTrans})$

AddHealthRecord(P). In order to add a patient's health record to the system, a new counter value must be obtained from R. Using this value, a doctor can create an identifier ID for the record. The record itself is then signed by a deanonymizable credential show and stored by $DBMan$ in the database.

The communication between D and both central authorities should be anonymous. Furthermore, for accountability reasons, D must prove to the registrar that she is a valid doctor knowing the private key $sk_p(i)$. This is done by a deanonymizable credential show combined with a GQ proof of knowledge.

1. D : $(sk_p(i), pk_p) = \text{retrieveKeypair}(P)$
2. $D \leftrightarrow R$: $\text{showCred}(\mathit{Cred}_d, \{\text{'doctor'}\}, \mathit{AddReqCond}, pk_p) \rightarrow \mathit{ShowTrans}$
3. $D \leftrightarrow R$: $\text{GQProof}\{RSA^{-i}_{(n,v)}(pk_p)\}$
4. R : set $n_i = n_i + 1$
5. $D \leftarrow R$: $\text{send}(n_i)$
6. D : create $ID = \mathcal{H}(sk_p(i) \parallel n_i)$
7. $D \leftrightarrow DBMan$: $\text{showCred}(\mathit{Cred}_d, \{\text{'doctor'}, [\text{specialties}]\}, \mathit{AddCond}, (ID, data)) \rightarrow \mathit{ShowTrans}_{ID} = \mathit{record}_{ID}$
8. $DBMan$: add record_{ID} to database

RetrieveHealthRecords(P). To retrieve all health records of a patient, D requests from R all counter values for all of the patient's current and previous private keys. Once these are retrieved, D can compute the corresponding record IDs and hence request P's records from the database.

Again, communication between D and the central authorities should be anonymous. Also, *DBMan* logs the retrieval transcripts for accountability purposes.

1.	D	: $(sk_p(i), pk_p)$ =retrieveKeypair(P)
2.	$D \rightarrow R$	: indexRequest(pk_p)
3.	$D \leftarrow R$	: send($\{1, n_1\}, \ldots, \{i, n_i\}$)
repeat step 4: $\forall j \in \{0, \ldots, i-1\}, \forall k \in \{1, \ldots, n_{(i-j)}\}$:		
4.1.	D	: create $ID_{jk} = \mathcal{H}(\mathrm{RSA}^{j}_{(n,v)}(sk_p(i)) \parallel k)$
4.2.	$D \leftrightarrow DBMan$	: showCred($Cred_d$, {'doctor'}, $RetrCond$, ID_{jk}) → $ShowTrans$
4.3.	$D \leftarrow DBMan$	: send($record_{ID_{jk}}$)
4.4.	$DBMan$	: log($ShowTrans$)

ChangePrivateKey. A patient changing her temporal key $sk_p(i)$ into $sk_p(i+1)$, reports this change to the registrar. She provides R with a verifiable encryption of her new private key and retrieves a credential $Cred_{sk_p(i+1)}$ for her new secret key. In addition, the patient's old credential is revoked.

1.	$P \leftrightarrow R$	: showCred($Cred_{sk_p(i)}$, {'patient', P, pk_p, i}, $null$, $null$) → $ShowTrans$
2.	$P \leftrightarrow R$	: $\omega_{sk_p(i+1)} = \mathrm{VEncrypt}_{pk_{es}}\{sk_p(i+1), sk_p(i+1) = RSA^{-(i+1)}_{(n,v)}(pk_p)\}$
3.	R	: revokeCred($GetTrans_{sk_p(i)}$)
4.	$P \leftrightarrow R$	: getCred({'patient', $P, pk_p, (i+1)$}) → $Cred_{sk_p(i+1)}$; $GetTrans_{sk_p(i+1)}$
5.	R	: replace $\{i, \omega_{sk_p(i)}, GetTrans_{sk_p(i)}\}$ with $\{(i+1), \omega_{sk_p(i+1)}, GetTrans_{sk_p(i+1)}\}$, append $\{(i+1), n_{(i+1)} = 0\}$ to stored data

EmergencyRetrieval(P). An emergency doctor *ED* may in emergencies request the patient's private key. This is done by anonymously filing a deanonymizable request with the emergency service *ES*. By decrypting the verifiable encryption of $sk_p(i)$, *ES* can recover the patient's private key.

1.	$ED \leftrightarrow ES$	: showCred($Cred_d$, {'doctor','ER'}, $ERCond$, {$P, motivation$}) → $ShowTrans_{ED}$
2.	ES	: evaluate and store request
3.	$ES \leftrightarrow R$	: request($\omega_{sk_p(i)}$)
4.	ES	: $sk_p(i) = \mathrm{decrypt}_{sk_{es}}(\omega_{sk_p(i)})$
5.	$ED \leftarrow ES$	: send($sk_p(i)$)

4 Evaluation

Both patients as well as doctors have privacy concerns regarding electronic health records. First of all, patients do not want their medical history to be publicly

available. Also, in order to maintain their autonomy, doctors do not want their activities to be centrally trackable.

4.1 Privacy Control for the Patient

Because of the strong hash function $\mathcal{H}$ and secret input $sk_p(i)$, different health records belonging to the same patient are unlinkable. As a consequence, only authorized doctors being in possession of $sk_p(i)$ can access and link the patient's medical information. As these doctors know $sk_p(i)$, they must either enjoy the patient's trust, or be working at an emergency room.

By updating her private key, a patient prohibits all doctors from adding or retrieving any of her new health records. She can then renew her trust relation with some of these doctors by providing them with her new private key. All other doctors, however, will no longer be able to manage the patient's new health records. A key update is executed, for example, when a patient changes doctors or after an emergency retrieval.

Timing analysis may allow the database manager to estimate linkabilities between health records. To solve this problem, a doctor should not retrieve all of her patient's health records at once. Also, it is advisable to use anonymous communication channels and to store a cache of previously retrieved health records.

In case of an emergency, the emergency service *ES* can recover a patient's private key $sk_p(i)$. Hence, *ES* can retrieve and link all of the patient's health records. This is necessary to allow for a good service, for example when the patient is unconscious and her regular doctor is not available. However, the service must be trusted not to abuse her recovering powers. In order to minimize this trust, arbiters can inspect the recovery process. Also, trust can be distributed over multiple emergency services, who then have to cooperate to retrieve $sk_p(i)$.

When a patient detects abuse such as unauthorized access to her health records or the addition of wrong information, she can file a complaint. The doctor responsible for the abuse can then be identified and appropriate actions can be taken. (e.g. the doctor's credential could be revoked)

Although a patient can decide which doctor to trust, she cannot hide any medical information from this doctor. Indeed, a doctor can only accept a patient's trust, if she is shown a valid credential containing the patient's current private key $sk_p(i)$.

4.2 Autonomy of the Doctor

A health record in a database is actually a transcript *ShowTrans* of an anonymous credential. Therefore, the record does not reveal anything more about its creating doctor than her status as an authorized doctor with the specified specialties

Apart from the doctor registration procedure, all communication between a doctor and the central authorities (*R,DBMan* and *ES*) is anonymous. Hence, the only doctor information known to a central authority, is whether or not this doctor is registered with the system.

The anonymity received by a doctor is conditional, and can be revoked if abuse is detected. This revocation is performed by a third party trusted not to perform arbitrary deanonymizations. This trust can be minimized by using arbiters and by distributing the power to deanonymize over multiple organizations.

A doctor might want to know the identity of a health record creator. This is useful when she wants to share advice or when she needs extra information about the patient. Doctors can achieve this by anonymously filing a deanonymizable request to a trusted deanonymization organization. An alternative is the use of health records which contain no medical data but a reference to, and access information for another database. This may for example be a hospital's database containing all health records created by doctors affiliated with this hospital.

4.3 Scalability

An important issue when regarding a practical implementation is the scalability of the system. If more people use the system, a shift from a single central database towards multiple databases will be necessary. The registrar will then need to keep extra bookkeeping information about where each record is situated. Another potential problem is the possibility for collisions of hashfunctions. This can be countered by using multiple hashfunctions in order to create a unique record *ID*. Also, multiple RSA instances (n, v) can be used.

5 Related Work

The Health Insurance Portability and Accountability Act [9] imposes the development of national standards for electronic healthcare transactions. Next to this, it states strong requirements concerning security and data protection safeguards for medical information. The most important of these security safeguards is access control. The first proposals to solve this issue made use of public key infrastructures (PKIs). However, PKI technology was not designed for implementing access control. Rather, it was designed for public key cryptosystems to provide for confidentiality and integrity protection of data, and authentication of users. This authentication property can be used to implement access control. However, as each certificate is unconditionally linked to a (possibly pseudonymous) identity, all the user's transactional data can be collected. This has devastating consequences for user privacy.

Another solution is role based access control [12, 6] in combination with anonymous communication [5, 10]. It enables access control based on contextual information rather than on identity. However, anonymity is unconditional and abusive behaviour cannot be punished.

To allow for patient control, a shift towards patient-involvement, for example by the use of smartcards [8], is required. Such a shift allows the patient to view and control her own information. A complete shift is undesirable though, as this would allow the patient to add, delete or modify her own information.

6 Conclusions and Future Work

In this paper we have described a secure and privacy-preserving electronic health record system. The system protects the patient's privacy by allowing her to control who has access to her medical information. However, no personal information can be hidden from the medical practitioner entrusted with this access. Furthermore, the doctor's anonymity is conditionally preserved.

Future work includes research on how to combine the system with smartcard technology. These smartcards could contain, for example, the patient's private keys or medical certificates stating her blood group or a particular disease. Note though, that our setting requires the private keys to leave the smartcard, which is an alteration of the traditional smartcard setting.

Other work includes the usage of health records for statistical analysis. Such usage will require a transformation from the original database without linkabilities to a new anonymized database with linkabilities. Finally, a framework for handling disputes and abuses will be developed.

References

1. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures (extended abstract). In *EUROCRYPT*, pages 591–606, 1998.
2. S. A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy.* MIT Press, Cambridge, MA, USA, 2000.
3. J. Camenisch and I. Damgård. Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes. In T. Okamoto, editor, *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 331–345. Springer, 2000.
4. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT*, pages 93–118, 2001.
5. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. In *Communications of the ACM*, volume 24, pages 84–88, February 1981.
6. D. F. Ferraiolo, J. F. Barkley, and D. R. Kuhn. A role-based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information Systems Security*, 2(1):34–64, 1999.
7. L. C. Guillou and J-J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both trasmission and memory. In *EUROCRYPT*, pages 123–128, 1988.
8. Health card technologies. `http://www.hct.com`.
9. Health insurance portability and accountability act. `http://www.hipaa.org/`.
10. A. Pfitzmann and M. Waidner. Networks without user observability: Design options. In *Advances in Cryptology- EUROCRYPT '85: Proceedings of a Workshop on the Theory and Application of Cryptographic Techniques*, pages 245–253, 1985.
11. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.
12. R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.
13. M. Stadler. Publicly verifiable secret sharing. In *EUROCRYPT*, pages 190–199, 1996.