Security in Wireless Medical Networks

Eskeland, Sigurd

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

Security in Wireless Medical Networks

Ph.D. Thesis

by

Sigurd Eskeland

2008

Submitted to

Department of Communication Technology,
Faculty of Engineering and Science,
Aalborg University, Denmark

Supervised by

Prof. Vladimir Oleshchuk, University of Agder, Norway
Prof. Neeli Prasad, Aalborg University, Denmark

Security in Wireless Medical Networks

(Sikkerhed i Trådløse Medicinske Netværk)

by Sigurd Eskeland, 2008

This project was funded by
The Research Counsil of Norway
Grant no. 153935/V50.

ISSN: 0908-1224
ISBN: 87-92078-45-1

Abstract

In the medical scenario, electronic information management and wireless computer networks provide ubiquitous access possibilities to medical databases which may comprise hundreds of thousands of electronic patient records (EPRs). Such records may contain personal and highly sensitive patient data, and it is therefore necessary to limit the accessibility of such data to only concerning medical personnel. Criteria for granting (or authorization) of EPR access should be based on legitimacy and the need-to-know principle, meaning that only medical personnel that is going to provide medical care to a given patient should be granted access to the necessary medical data of the concerning patient they are going to provide care for.

EPR access control based on such criteria would help to prevent illegitimate persons from accessing patient data. This could be implemented by means of patient consent, where EPR access is provided to legitimate medical practitioners due to the consent of the pertaining patient. It could also be implemented by group consensus, where the consensus of a minimum number of concerned participants could qualify for authorization of EPR access. Hence, a given number of associated medical practitioners, e.g., a medical team, could be recognized as a proper basis for trust. Since medical care is often provided by medical teams, the consensus of a minimum number of the team members would thus act as a qualifying criterion for such teams to acquire EPR access.

The hierarchical ranking of medical practitioners is significant, since higher ranking implies more privileges. Medical practitioners of higher ranking (e.g., medical doctors) are accordingly in position to be entrusted access to more sensitive medical data than practitioners of lower ranking (e.g., nurses). An essential hierarchical security requirement is to ensure that the medical practitioners are only granted access to EPR data whose confidentiality level is in agreement with the user ranking, and to EPR data associated with the underlying confidentiality levels.

In addition to the mentioned accessibility issues, protection of communicated data to and among multiple users is likewise of essential interest in the medical scenario. In this thesis, we present a number of cryptographic methods for secure establishment of EPR access according to the mentioned issues, and methods for secure transmission of medical data over insecure wireless networks.

Contents

Part I

Introduction

Chapter 1

Introduction

As the use of information technology has progressed in health care, there has been an increased focus on security and confidentiality issues of electronic patient records (EPR) in medical environments, and the need for secure and confidential management, handling and storage of such [3, 6–8]. Electronic information management and computer networks enable ubiquitous access possibilities to medical databases which may comprise hundreds of thousands of electronic patient records (EPRs). Such records may contain personal and highly sensitive patient data, which could include sensitive data about AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, sexual divergencies, genetic predispositions to diseases, information about toxic addictions, and so on [6]. It is therefore necessary to limit the accessibility of such data to concerning medical personnel only. Since there would be a continuously flow of patients being hospitalized, access cannot be based on long-term, predefined permission assignments between medical practitioners and EPRs, but must rather be established dynamically according to needs.

Two important issues in this context concern proper authorization of EPR access and secure communication, including secure group communication since the group aspect plays an important role in the medical scenario. We consider the need-to-know principle as a criterion for authorization of EPR access, since only medical practitioners providing medical care to a given patient (or patients) should be granted only access to the necessary medical data of the concerning patient they are providing care to. Concerning what type of personnel should be in charge of granting EPR access, we consider 1) patients, 2) security administrators, and 3) medical personnel.

It has become a widely recognized principle that patients have a right to exert control over their own medical data [1, 2, 4, 5]. That is, EPR access authorization carried out due to patient consent. Alternatively, security administrators acting as a trusted third party could grant medical personnel access to the EPR of a patient they are going to provide care to on behalf of the patient. This would particularly be relevant in emergency situations where the patient could be unconscious.

Since medical care is often given by practitioners in medical teams working for the same cause, the group aspect can by itself be recognized as a proper basis for trust. Group consensus could conveniently act a qualifying criterion for a minimum number of associated medical practitioners to acquire access to a given EPR. Enforcement

of such an requirement, for instance by means of a threshold cryptosystem, would conveniently prevent single individuals from illegitimately accessing patient data.

In this thesis, we present several cryptographic schemes for EPR authorization based on group consensus and patient consent, and which are well-suitable for wireless networks. There are two schemes for EPR authorization based on patient consent. An essential feature of the first is that it provides secure storage of medical data due to that each EPR is encrypted on a long-term basis with a unique key on the EPR server. There is no accompanying table of cryptokeys. By means of the partial computations of the consenting patient and the medical practitioners of the granted team, relevant EPR cryptokeys are securely reconstructed on the EPR server out of reach of any of the involved participants. The EPR cryptokeys enable subsequent decryption of the EPRs of the granting patients. The second patient consent-based scheme provides patient anonymity. It could in some cases be desirable to hide associations between meaningful identifies such as patient names, social security numbers, etc., and the patient record. This could especially be desirable in cases of celebrities, politicians and for particularly sensitive data, like AIDS/HIV status, etc. The scheme thus allows patients to be in charge of granting EPR access while still remaining anonymous.

The hierarchical ranking of medical practitioners is also of significance, since higher ranking implies more privileges and that access to information of high sensitivity could be entrusted. For example, doctors would be hierarchically ranked higher than nurses, and would reasonably be entrusted access to more sensitive data. Since the medical practitioners are hierarchically ranked according to their job positions, user hierarchies would play an essential role regarding what kind of information that practitioners should be granted access to.

The thesis also includes a number of general-purpose cryptographic protocols for secure group communication with applicability for wireless medical networks and the medical scenario. For example, a hierarchical key establishment scheme is proposed for secure group communication and data distribution. It prevents participants of a given ranking to access information (encrypted with a pertaining hierarchical session key) that is classified above their ranking, while they can obtain information pertaining to their own and lower levels.

Chapter 2

Preliminaries to Information Security in the Medical Scenario

2.1 Introduction

With the emergence of information technology in health care, there has been much focus on security and confidentiality issues of electronic patient records (EPR) in medical environments. Medical records contain confidential personal information which could even include sensitive data about AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, sexual divergences, genetic predispositions to diseases, information about toxic addictions, and so on [6]. Due to the sensitivity of medical data, it is therefore essential that such information is protected from disclosure except when medical practitioners require access to patient records in order to provide medical care to patients. An important issue in this context concerns proper establishment (i.e., authorization) of EPR access. A basic criterion for this should be legitimacy, meaning that only medical personnel going to provide care to a given patient (or patients) should only be granted access to the necessary medical data of the concerning patient they will providing care to. Another significant security issue concerns secure and confidential management, handling, transport and storage of personal medical information [8].

Security of medical networks and preservation of privacy of medical data have for one or two decades been topics of concern and scrutiny, since almost every person would have at least one patient record containing personal and confidential medical information. Since the manual record keeping systems of the past lacked automatic enforcement of access control, medical practitioners would necessarily not be prevented to access arbitrary patient records. Thus, the confidentiality of patients was resting considerably on the discretion of each individual medical practitioner and legal enforcement.

Today, medical data is in general managed by networked computer systems which have replaced paper-based patient records and manual record keeping systems. Health organizations and hospitals are administrating large databases of such personal electronic patient records. Computerized medical databases have a number of advantages compared to paper-based systems concerning flexibility, functionality

and a more effective data management due to the possibly ubiquitous accessibility of data, independently of location and time. This complies well with decentralized organizations since data can be easily transferred within and across health establishments by means of wired and wireless computer networks.

In agreement with common medical ethics and due to the confidential nature of medical data, access to medical data should rest on the basis of need-to-know and legitimacy. In other words, the medical data of a patient should only be disclosed to medical personnel that has a legitimate need to access the medical data of the patient, in order to provide proper medical care to that patient. Proper measures should be taken to confine the accessibility of the data in agreement with the "need-to know"-principle. The large amounts of medical data ubiquitously accessible due to computerized data management and networking raise important needs and requirements concerning the security and privacy of medical data and confidentiality of patients.

Threats and violations to the privacy medical data could just as well come from within the health organization than from outside, and it is essential that proper access control mechanisms and data protection should be facilitated. In this chapter, we discuss relevant security and privacy aspects that are investigated in this thesis, and also solutions for and enforcements of such. Issues of interest include among others authorization and granting of EPR access, patient consent, EPR access acquisition, teams, user hierarchy, and other related issues.

2.2 Patient confidentiality

The necessity to protect patients' privacy is a focal point of importance. The same goes for the importance of protecting patients' medical records that may contain very sensitive personal information as noted. Electronic medical database systems and networking may provide efficient data management and opportunities for ubiquitous data accessibility, which may create needs for strengthening ethical and legal requirements correspondingly.

2.2.1 Patients' rights

A significant factor related to patient confidentiality is the right the patient has to decide the course of action to be undertaken in regard to the medical practitioners. With respect to patient confidentiality and patient consent, American Medical Association (AMA) [1] states that

> the physician's duty to maintain confidentiality means that a physician may not disclose any medical information revealed by a patient or discovered by a physician in connection with the treatment of a patient... The physician generally should not reveal confidential communications or information without the patient's express consent unless required to disclose the information by law.

Patients' rights for self-determination may include:

1. An informed basis for medical treatment and medical procedures to be undertaken.
2. An informed basis of who is requesting access to the EPR of the patient, and having the right to grant or deny it.
3. An informed basis of the motivations and purpose for medical and scientific research that involves access to medical data, including the identities of the individuals undertaking the inquiry.

In the latter case, patient anonymity would be useful to preserve the confidentiality of concerning patients. By means of anonymization, public identifiers such as names, addresses, social security numbers, etc., would be hidden and replaced by pseudonyms that can be randomly selected, e.g., random numbers. Pseudonyms can be issued on a long-term basis, but a long-term association between a pseudonym and an EPR would lower its effectiveness in the long run. Temporary or session-wise pseudonyms would on the other hand effectively prevent that the association between an EPR and a pseudonyms can be discovered over time.

2.2.2 Issues relevant to patient confidentiality

A security system with proper access control facilities would be required to ensure a minimum level of patient confidentiality. Access control would, however, raise questions such as what would be the criteria for access to patient records to be granted? Who should grant access? Should the patient be able to influence how and to whom access to his or her EPR is to be granted? In what way should user roles or the job position of the user, for example whether the user is a nurse, doctor or specialist, infer restrictions on what parts or data modules of a granted EPR that the user may be allowed to access? Should user roles impose any restrictions on operations (read, write, etc.) to be employed on the granted data? How should access control be managed regarding medical teams? Concerns should also be taken about the confidentiality of stored medical data, and about the confidentiality and integrity of data that is in transfer over networks.

2.3 Players in the medical scenario

In this thesis, we recognize the main players in the medical scenario to be as follows:

- The owner of the EPR, i.e., the patient. Each patient is represented by one EPR that he or she imposes ownership onto.
- Medical practitioners like doctors and nurses. Two or more medical practitioners can be associated, forming medical teams. We distinguish between ad hoc (short-term) medical teams, and long-term medical teams.
- Information security administrators, whose role as a trusted third party could be to provide and assign required credentials (i.e., assign user roles and provide corresponding long-term user keys) to the main players. Another function could be to grant medical practitioners access to relevant data.

The medical practitioners providing care to a patient are to be dynamically assigned or granted access to the electronic patient record of this patient. Such assignments form short-term, temporary relationships between a given EPR and the pertaining medical practitioners, whereof the duration would be in accordance to the period of treatment.

2.4 Authorization and granting of EPR access

As previously noted, the need-to-know principle is a proper criterion for authorization of EPR access to medical teams, since this ensures that only those who provide medical care to a certain patient are to be granted access to this patient's medical data. Although given such a criterion, somebody has to be in position of granting medical practitioners access to relevant medical data.

The principle of separation of duty requires a minimum number of participants to carry out some types of tasks or transactions in order to prevent fraud or errors [9]. Enforcement of separation of duty would thus prevent individuals to carry out such tasks or transactions on their own. Typical applications for this principle can be found in banking, where for example two employees, one manager and one clerk, could be required to carry out some financial transaction. Access to the bank vault is also a good example. It may not be desirable that individuals could solely access the vault due to the risk of fraud, robbery and extortion. The safety would be considerably improved by a threshold-oriented security system, where the participation of at least 2 or 3 arbitrary persons out of for instance 4, each holding a unique and secret key, is required in order to unlock the vault.

This is an example where the consensus of a specific minimum number of associated people is required to carry out the action. Threshold-oriented cryptosystems are suitable to enforce requirements of group consensus, since such cryptosystems require the participation of minimum t of n associated users in order to carry out a cryptographic computation. Enforcing a threshold requirement can be an effective way to prevent fraud or other actions that could be in violation to a security policy. The threshold requirement is a special case of the separation of duty principle. Examples of such cryptosystems can be found in [20–26].

In this thesis, we consider the consensus of a minimum number of relevant participants to be a useful criterion for authorizing access to medical data. We consider patients, medical practitioners and security administrators to be relevant to be in position of authorizing EPR access. This yields the following cases:

- Medical practitioners or a medical team can obtain access to a relevant EPR as a function of the consent of the concerning patient to whom they are going to provide medical care.

- Medical practitioners or a medical team can obtain access to a relevant EPR due to the consensus of a minimum number of security administrators.

- A medical team can obtain access to a relevant EPR due to the consensus of a minimum number of its members.

In the following subsections, we discuss the function of consensus of each of these three types of players as a criterion for to grant or acquire EPR access.

2.4.1 Patient consent

EPR access could be granted based on patient consent. This has become an important principle in medical ethics, since it has be broadly recognized that patients have a right to exert control over whom is to access their medical data. It is therefore reasonable that patients should decide who that should (and should not) access their EPR by means of consent, and thereby exert control over their respective EPRs. This would permit the patient to grant specific medical personnel access his or her EPR. This has in the past been implemented manually by means of written consent from the patient in many health establishments. Such manual practice would rely in individuals discretion, and would have no automatic control enforcement. In electronic patient record data systems, patient consent could be securely enforced by means of an automatic security system and cryptographic methods. Access control schemes based on patient consent are proposed in Paper G and I. Also see [2, 4, 5] for further discussion.

2.4.2 Trusted third parties

Security administrators are trusted parties that may authorize medical personnel access to EPRs on behalf of patients. However, such an arrangement would entail that the administrators may have access to all patient records at that given institution, or even less desirable, to a number of associated institutions due to networking. Such unconditional access to highly confidential information of a large number of people is in general an undesirable situation.

This possible security weakness would be minimized by requiring group consensus by means of a threshold-based cryptosystem. Such a cryptosystem would be suitable for enforcing the consensus requirement of two or more security administrators for providing EPR access to specific medical practitioners on behalf of the pertaining patients.

It should be noted that an EPR access granting scheme that is based on patient consent as discussed in the previous subsection, should also support the possibility of EPR granting by a trusted third party due to the emergency case. In emergency situations, patients may be unconscious and are thus unable to actively consent to EPR authorization. EPR access authorization carried out by security administrators, possibly threshold-based, could thus be an appropriate alternative to ensure patient confidentiality. Paper G includes a group-oriented scheme enabling secure EPR access granting by trusted third parties for the emergency case.

2.4.3 Data access acquisition

Medical care is typically provided by medical personnel organized in teams. Since a team consists of a number of associated medical practitioners working for the same cause, this by itself could be recognized as a proper basis for trust. Group consensus could thus be recognized as a qualifying criterion for medical personnel to acquire

EPR access. This criterion could conveniently be enforced by means of an appropriate threshold cryptosystem. Enforcement of such a requirement would accordingly prevent that individuals may snoop and arbitrarily read personal medical data unless holding special privileges. A team-oriented EPR access acquisition scheme is proposed in Paper H.

2.4.4 Threshold-oriented cryptosystems

Threshold secret sharing is a cryptographic approach that provides enforcement of group consensus requirements. According to this method, a secret number (for example, a cryptographic key) is split into n unique user "shares". One such share is handed to one person so that in a group of n persons, each is holding a share. The secret number can only be computed by means of at least t arbitrary such shares, where $t-1$ or less user shares reveal nothing of the secret number. The term *threshold* denotes the minimum number of participants of the group whose user shares are required to compute the secret number. Threshold-orientation provides flexibility in contrast to that all n participants, i.e., a fixed set of participants, are required to carry out such a computation, since one or more of the n participants could be unavailable or absent at the moment.

Threshold cryptosystems are relevant in scenarios where group consensus is required, for example that an originator of some sensitive information like a secret key, is only willing to let it be disclosed due to the consensus of a given number of the associated individuals. Accordingly, it is prevented that single individuals can obtain the secret on their own.

An instance of a threshold secret sharing scheme is basically useful only once, since the participants have to reveal their secret user shares to subsequently compute the shared secret number. And once this is computed and revealed, it is revealed once and for all. However, threshold secret sharing is in practice used as a building block in threshold-oriented cryptographic schemes, where typical applications are threshold decryption and threshold signatures. There also exists conference key agreement schemes based on threshold secret sharing (e.g., [27]). Note that in such schemes, the user share of each participant is protected from being revealed to the other participants.

Threshold cryptosystems are commonly based on the Shamir secret sharing scheme [30]. Threshold decryption is a class of such cryptosystems, where a group of n member is each given a secret user share. The group is represented by a corresponding public key which allows encryption of plaintexts. Decryptions can only be carried out due to the partial computations of t arbitrary group members, in agreement with the threshold requirement. Examples of such schemes are found in [20, 25, 26]. Threshold signatures (e.g., [23, 24]) is another class of threshold cryptosystems enforcing that t arbitrary participants are required to compute such signatures in agreement with the threshold requirement.

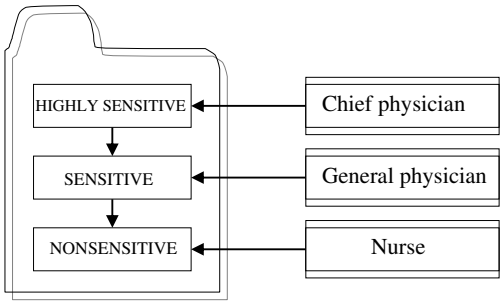

Figure 1: Example of a totally ordered hierarchy.

2.5 Hierarchical aspects

In medical organizations, medical practitioners and other employees are ranked according to their job positions in agreement with a hierarchical structure. Such a hierarchical organization implies that higher ranking correspond to more privileges and responsibilities. Medical practitioners of higher ranking (e.g., medical doctors) would accordingly be in position to be entrusted access to more confidential information than practitioners of lower ranking (e.g., nurses). Correspondingly, it is reasonable or even necessary that practitioners of lower rankings should be privileged access to less sensitive or confidential medical information than those of higher rankings.

Electronic patient record systems provide opportunities for fine-grained access control with regard to the sensitivity of the data. Depending on the desired level of fine-grainedness, each data field, data block or data module of the EPR, could be each assigned a confidentiality level (or sensitivity level) from a small range, for instance [0-3] where 0 denotes none or low confidentiality level and 3 denotes high. Alternatively, the confidentiality levels could be classified as open, nonsensitive, sensitive, and highly sensitive. Each data item could be individually marked according to its confidentiality level, or grouped into modules according to confidentiality level.

It is necessary to have a reasonable agreement between the user rankings and the confidentiality level of each blocks of an EPR. An essential security requirement would be that the medical practitioners should only be granted access to data whose confidentiality level is in agreement with the user ranking, and to data blocks of underlying confidentiality levels. It must accordingly be prevented that participants may obtain access to data whose confidentiality level is above the privilege level of that participant.

An example is showed in Figure 1, where each EPR is categorized into three confidentiality levels. There are three levels of user rankings; nurse, general physician and chief physician, that agree with the EPR confidentiality levels. The structure is a totally-ordered, meaning there is only one class of users for each level. Alternatively,

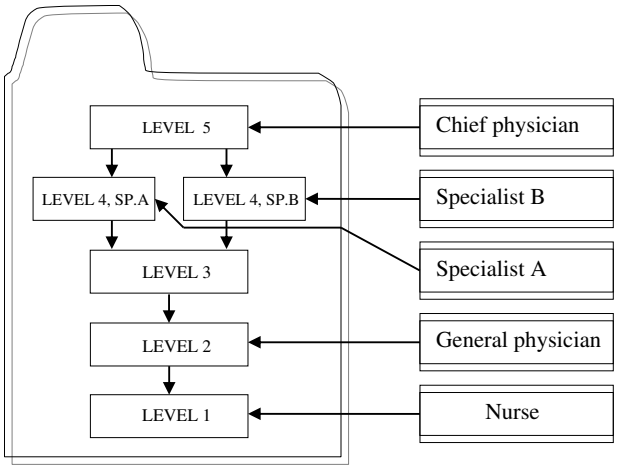

Figure 2: Example of a partially ordered hierarchy.

since there may be a number of types of medical specialists, compartmentalization of a given level into two or more user classes could be useful. An example is showed in Figure 2, where level 4 is compartmentalized into two classes. Accordingly, confidential data pertaining to one specialist area would not be accessible to specialists of other areas. Also note that chief physicians would have access to all of the EPR data.

Proper permissions concerning which data operators to be permitted used by the users in agreement with the confidentiality levels must be assigned. Typical permissions would be read, write and delete. Using Figure 2 as an example, the General Practitioners are assigned read, write for data level 2, and read, write, delete for data level 1.

A cryptographic EPR access control scheme based on patient consent for hierarchical medical teams is presented in Paper I. General hierarchical key establishment schemes are presented in Papers A and C.

2.6 Securing medical data

The EPR servers must authenticate users that are logging on before access to medical data can be provided. After a user has been successfully authenticated, access control has to be carried out in order for possible EPR access authorization to take place, meaning a given user must be validated to be granted access to the requested data or not. The access authorization would determine which data that the user could be allowed to access, and what operations that the user would be allowed to use. The server would then securely communicate the data to the user. Secure communication can be achieved by cryptographic key establishment protocols for secure establishment of secret session keys shared among the pertaining parties for subsequent encryption of the communicated data.

In this chapter, we point at the following for protecting data in a network:

- User authentication and access control.
- Preserving confidential communication between two entities.
- Preserving confidentiality of stored data.

2.6.1 User authentication

A user that is going to communicate with one or more users over an insecure network needs to make sure that the other users are really those who he or she thinks they are since they cannot see each other physically. Otherwise, an adversary may successfully masquerade as a peer user, and thereby obtain confidential information. A server needs to authenticate users logging on before confidential access can be provided. Likewise, a user needs to make sure the authenticity of a server when logging onto the server before supplying personal data to the server.

Secure user authentication can be achieved by means of cryptographic authentication protocols. Such protocols enable unilateral authentication; meaning authentication of one user towards another user, or mutual authentication; enabling both users to authenticate each other.

2.6.2 Access control

In the medical scenario, after a medical practitioner has been successfully authenticated by an EPR server, it has to validate the credentials of the person to decide whether access to the requested data can be granted or not. This would also include determining what operations that the user would be allowed to perform on the data. If the authentication and access control succeed, the subsequent communication should be protected.

2.6.3 Secure communication

It is essential that data communicated over computer networks are protected to prevent unauthorized individuals from eavesdropping, and to preserve the integrity of the communicated data. Secure communication is normally achieved by encryption, and normally requires the use of security protocols that have to provide relevant security properties. Cryptographic key establishment protocols provide secure establishment of session keys shared among the users that are going to securely communicate over the network by subsequent encryption of the communicated data. Many such protocols also provide user authentication.

Security protocols are designed with the assumption that the concerning computer networks are insecure, i.e., with the possible presence of an adversary. Such an adversary could be passive or active. This is a most reasonable assumption for wireless networks due to the broadcast-orientation, where any broadcasted message can be easily eavesdropped by anyone present. A passive adversary would have the ability to eavesdrop the communication over the network, whereas an active adversary would be able to change, replace or suppress data that is in transfer. A type of

active attacks is replay attacks where an adversary attempts to subvert a security protocol by replaying former messages. Measures like incorporating MACs or digital signatures should be taken to ensure that the integrity of data is intact.

The proposed security schemes in this thesis are designed for secure communication over wireless networks.

2.6.4 Secure data storage

It could be desirable to preserve a more overall level of information confidentiality, not only with respect to access control mechanisms and on what basis should granting of EPR access take place (that is, what should be the criteria and conditions for granting EPR access), but also concerning the long-term storage of the medical data.

It could be argued that long-term encryption of medical data may increase the overall security level of the information system than if stored as plaintext. This assumes that the corresponding cryptokeys would have to be "out of reach", since encryption imposes the problem of secure key management and key storage. If a cryptokey is compromised, its encrypted data would correspondingly be considered compromised. Storage and management of EPR cryptokeys would therefore impose a potential security risk since whoever controls such keys also controls the corresponding data. The subsequent storage and management of those cryptokeys would therefore directly reflect the actual achieved security.

The security schemes presented in Paper G and Paper H assume that the stored EPRs are encrypted by unique and distinct cryptokeys that are secret to all. Instead of employing tables containing EPR cryptokeys, these schemes enable secure temporarily establishment of an EPR cryptokey as a function of the computations of the users, and without revealing it to others than the EPR server.

2.7 Cryptographic building blocks

2.7.1 Secure binding of users and data

User-oriented security and data-oriented security are two relevant aspects related to the medical scenario:

- Data-oriented security includes preservation of secrecy and confidentiality of data, integrity of data, data authentication and non-repudiation of data.
- User-oriented security includes user authentication and ensuring proper access control.

Thirdly, secure association between users and data is perhaps of most interest here due to the protection of EPRs associated to individual patients, and the temporary associations between such data and legitimately medical practitioners.

This secure binding is established between the pertaining data and a personal long-term cryptographic key representing a user. Such binding is realized by encryption methods and digital signature methods, where such methods are based on

shared- and public key cryptography. Particularly the latter plays an important role in the methods presented in this thesis and in the recent literature in general.

2.7.2 Shared key cryptography

Symmetric key ciphers (i.e., shared key ciphers) have the advantage of relatively short keys and the ability of high rates of throughput. In such systems, secret cryptokeys must be shared among those entities that are to communicate confidentially, so regarding two-party communication, the key must remain secret in both ends. In large networks, there would be many keys to be managed, and each user would have to securely manage a list containing the key pairs for each of his or her contacts. This could be impractical and troublesome, and lack flexibility concerning new and leaving users.

Sharing long-term secret keys among a number of users is an impractical and troublesome assumption due to increased vulnerability from the aging of keys, and lack of flexible user constellations due to the shared keys. This could be mitigated by an online trusted third party (TTP) so that all communication goes through the TTP that shares a secret key pair with all relevant users. Nevertheless, this would in many cases be undesirable. It would correspondingly be a problem to distribute and establish new shared keys to new contacts over an insecure network if the key distribution protocol (that is, the key establishment protocol) is based on symmetric key ciphers, since this would require that a symmetric key is already shared between the distributor and the receiver.

In practice, symmetric key ciphers is mostly applied session-wise due to their capabilities of high throughput and efficiency. The shared session keys would be established by means of some secure key establishment protocol. Such protocols could be based on symmetric key ciphers or public key ciphers. However, a key establishment protocol based on symmetric key ciphers would still require that the two parties going to establish a shared secret session key still share a long-term secret key. Key establishment protocols based on public key ciphers eliminate the disadvantages of sharing long-term secret keys.

2.7.3 Public key cryptography

In public key systems, each user has a personal public/private key pair where the public key is publicly representing the pertaining user. This eliminates the disadvantage of sharing long-term secret keys. However, the downside of such systems would be lower throughput rates than symmetric key schemes, and relatively large key sizes.

Secure data transfer is achieved by means of encrypting a message with the public key of the receiver. By means of the corresponding private key, the receiver would be able to decrypt the pertaining cryptotext. The receiver would however not be able to authenticate the sender unless the sender provides a digital signature of the message. Examples of popular public key cryptosystems are RSA [29] and ElGamal [28].

The authenticity of public keys could be provided, so that it can be certified that a given public key is actually representing the users that it is claimed to do.

Otherwise, an adversary could substitute such a key with his or her own public key, and then be capable to subsequently decrypt confidential information that was intended for another person. Digital certificates are a means for establishing the authenticity of public keys. Digital certificates basically contain the identity and the public key of the user, and a digital signature of a TTP, typically the party issuing the public key. By means of the digital signature, the public key can be subsequently verified.

Due to the inefficiency of such cryptosystems, a prior agreement of a secret shared session key would be more adequate for communication of multiple messages. Key establishment protocols based on public key ciphers are significantly more preferable than those based on shared keys since they do not, for instance, require an online third party. An example of a popular two-party key agreement protocol that is based on public key cryptography is the Diffie-Hellman protocol [31]. It does not provide user authentication, but is used as a building block in many other cryptographic protocols.

2.8 Cryptographic protocols

Cryptographic protocols comprise a wide range of methods for provision of secure computation to two or more participants connected to a computer network. More precisely, a cryptographic protocol is a distributed algorithm defined by a sequence of steps precisely specifying the actions required of two or more entities to achieve a specific security objective [15, p. 33]. Such actions would require use of cryptographic primitives. There are cryptographic protocols for secure user authentication, secure key establishment, secure voting, electronic cash, anonymous transaction schemes and so on. See for example [13, 14, 16, 17] for overviews.

Many times, the objective for data security also includes user security. For example in the case of secure communication where two users would like to communicate securely over an insecure network, each of them would first have to make sure that the other party is who he or she claims to be. Since the parties cannot see each other physically, user authentication has to be performed over the insecure network. Secondly, the parties would have to agree on some secret shared cryptographic session key for subsequent secure communication. Most practical user authentication schemes and key agreement schemes with user authentication are based on public key cryptosystems in contrast to symmetric key cryptosystems.

Due to the necessity of preserving patient confidentiality, such security properties of cryptographic protocols are mostly relevant for the medical scenario. This thesis includes secure methods for many of the aspects that are related to patient confidentiality as already mentioned. This includes aspects concerning the three types of players (patients, medical practitioners, security administrators), group orientation, hierarchical aspects, access control and EPR access granting, patient consent, patient anonymity, secure communication and key establishment, hierarchical access control issues, secure data management and storage, and more.

2.8.1 Security issues

The purpose of cryptographic protocols is to provide secure communication on insecure computer networks. Wireless networks are regarded insecure in the sense that communicated messages could easily be eavesdropped by a adversary (passive attack), or even modified or replaced (active attack). In general, an active adversary could be capable to suppress, replace, replay and modify messages over the network.

Cryptographic protocols have to ensure that two-party or multi-party computations can be carried out in agreement with some given security requirements. This normally includes preservation of confidentiality and privacy; preventing other than the participants from obtaining the inputs (e.g., the cryptographic user keys) and the computed results (e.g., a secret session key).

2.8.2 Secure key establishment

Secure communication over computer networks is usually achieved by means of encrypting the exchanged messages. The messages could be encrypted by means of long-term public keys (or long-term shared keys). However, the last case would require that they share the same secret key which can be achieved by means of some secure key establishment protocol.

By means of such protocols, two or more individuals can establish shared secret cryptographic keys over insecure networks. The protocols can be based on secret key cryptography or public key cryptography. Due to sharing of long-term secret keys among a number of users is an impractical assumption, most key establishment protocols that is based on shared key (a.k.a. symmetric key) cryptography require an online TTP. Hence, each user would share a secret key with the TTP, and all key establishment messages would go through the TTP. Kerberos [19] and the symmetric key protocol of Needham-Schroeder [18] are two well-known examples of key establishment protocols based on shared secret keys.

As noted in Chapter 2.7.2, it would be a problem to distribute and establish new shared keys to new users over an insecure network if a key is not already shared between the new user and the TTP. The advantage of public key ciphers is simplification of key management and eliminating the need for an online TTP. This increases considerably the usability for protocols based on public key ciphers, which have therefore become far more important than symmetric key protocols. Most public key protocols are based on a few well-known problems in number theory like the Discrete Logarithm Problem, the closely related Diffie-Hellman Problem, and the Factorization Problem (i.e., the difficulty of factorizing integers composed of two very large primes). For example, the RSA public key cryptosystem [29] is based on the Factorization Problem, and the ELGamal public key cryptosystem [28] is based on the two closely related Diffie-Hellman Problem and the Discrete Logarithm Problem [15, p. 294]. All security protocols in this thesis are public key-based.

Key establishment protocols can basically be divided into *key transfer protocols* and *key agreement protocols*. Key transfer is where one entity generates the secret key and distributes it confidentially to one or more users. Key agreement is where two or more participants that "agree" on a secret key by equally contributing to the

value of the established key. According to the number of participants, such protocols are categorized as two-party and multi-party protocols.

2.8.3 Group-oriented cryptographic protocols

Secure group communication refers to the scenario in which a group of participants can communicate securely over some computer network in such a way that the exchanged messages would be unintelligible for outsiders and non-pertaining users. Conference key establishment protocols (also known as multi-party key establishment protocols) allow a number of users to establish a shared session key whereof secure communication over insecure computer networks can be achieved by encrypting the exchanged messages. Group-oriented key agreement is a special case of *secure multi-party computation*, where n participants, $\mathcal{U} = \{P_1, \ldots, P_n\}$, compute the result of some function $f(x_1, \ldots, x_n)$ and where each $P_j \in \mathcal{U}$ holds a secret input x_j. The problem is how to compute f without revealing their secret inputs to any other party, including the other participants. The function could be any function taking any inputs where the computations are conducted over a distributed network. No information about the inputs should be learned from the computations. Strictly speaking, any participant that is legitimately included in the execution of a security protocol should not learn about the private inputs of the other participant. All that should be learned is the result of the protocol, in agreement to whom the result is designated to.[1]

In addition to key transfer and key agreement, such protocols can also be classified according to the nature of the group composition, i.e., according to whether the composition is ad hoc or predefined (long-term).

In Chapter 2.5, we pointed out the hierarchical aspects in the medical context, including hierarchical user ranking of the medical personnel, and that each data module of patient records can be assigned a confidentiality level in agreement with the user hierarchy. For this to be meaningful, hierarchical access control must be carried out, preventing that users can obtain access to data whose sensitivity level is above the corresponding hierarchical level of that user.

A relatively large class of hierarchical cryptographic schemes is known as Hierarchical Access Control. The main disadvantage of Hierarchical Access Control schemes is that such schemes basically provide computation of long-term, predefined hierarchical keys. This means that the new keys have to be distributed for every session from the key center. In contrast, secure hierarchical group communication could be achieved by means of hierarchical key establishment protocols. Such protocols facilitate secure establishment of a number of session keys in agreement with the given number of user levels. An essential security property is that users of a given level can compute the hierarchical session keys pertaining to their own and underlying security levels, while it is computationally infeasible to compute hierarchical session keys of overlying security levels.

A hierarchical key establishment protocol is presented in Paper A and Paper I, although the key establishment protocols in the literature are generally non-

[1] Most protocols distributes the same result for all participants, i.e., the participants collaboratively compute one result that is to be shared among them.

hierarchical.

2.8.4 Cryptographic access control

It is essential that only legitimate medical practitioners should be able to obtain access to relevant EPRs. An important issue is concerning authorization of EPR access, how this should be carried out, and who should be the authorizing parties. In this thesis, we present cryptographic schemes where EPR authorization is based on patient consent, the consensus of a minimum number of security administrators, and the consensus of a minimum number of the members of a medical team that requires access to a certain EPR. These are presented in Papers F-I.

2.8.5 Patient anonymity

Due to the sensitive nature of certain kinds of medical information, like information about AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, genetic predispositions to diseases, toxic addictions, etc., it may be desirable in such cases that identities (names, addresses, personal security numbers, etc.) of patients are hidden [10–12]. This is to prevent, for example, a person's name from being associated with possible physical disabilities of a sensitive nature. A common way of obtaining anonymity is by means of pseudonyms. A pseudonym is an identifier that could be a random number, and can be established for a long-term or temporary basis. A temporary pseudonym is preferred since it would prevent that association between a fixed pseudonym and the medical data of the pertaining patient can build up over time. There should be no deducible association between actual identities and patient pseudonyms. A cryptographic access control based on patient consent that provides patient anonymity is presented in Paper F.

Chapter 3

Contributions

This thesis consists of the following research papers:

Paper A Sigurd Eskeland, Vladimir Oleshchuk. Hierarchical Multi-Party Key Establishment for Wireless Networks. Journal of Information Assurance and Security (JIAS), No. 1, Vol. 3, Dynamic Publishers, USA, 2008.

Paper B Sigurd Eskeland. Public Group Key Cryptography. Tatra Mountains Mathematical Publications. Volume 37, No. 3, pp. 23-33, Mathematical Institute, Slovak Academy of Sciences, 2007.

Paper C Sigurd Eskeland, Vladimir Oleshchuk. Efficient Hierarchical Group-Oriented Key Establishment and Decryption. Proceedings of the Fourth International Conference on Information Assurance and Security (IAS), September 2008, Naples, Italy, pp. 67-72, IEEE Computer Society, 2008.

Paper D Sigurd Eskeland, Vladimir Oleshchuk. Collusion-resistant Threshold Decryption. Proceedings of IASTED 4th int. conf. on Communication, Network and Information Security (CNIS 07), Berkeley, CA, USA, pp. 12-15, ACTA Press, 2007.

Paper E Sigurd Eskeland, Vladimir Oleshchuk. Collusion-Resistant Threshold Cryptosystems. Proceedings of International Workshop on Coding and Cryptography 2009 (WCC 09), May 2009, Ullensvang, Norway.

Paper F Sigurd Eskeland, Vladimir Oleshchuk. Anonymity Authorization Granting in Medical Information Networks. Proceedings of European Conference on eHealth 2006, Fribourg, Switzerland, Lecture Notes in Informatics, Vol. P-91, pp. 71-81, Gesellshcaft für Informatik, 2006.

Paper G Sigurd Eskeland, Vladimir Oleshchuk. EPR Access Authorization of Medical Team Based on Patient Consent. Proceedings of European Conference on eHealth 2007, Oldenburg, Germany, Lecture Notes in Informatics, Vol. P-118, pp. 11-22, Gesellshcaft für Informatik, 2007.

Paper H Sigurd Eskeland, Vladimir Oleshchuk. Secure Team-Based EPR Access Acquisition in Wireless Networks. Proceedings of the Third International Conference on Availability, Security, and Reliability: Workshop on Security and Privacy in e-Health (WSPE 08), Barcelona, Spain, pp. 943-949, IEEE Computer Society, 2008.

Paper I Sigurd Eskeland, Neeli Prasad. A Novel Decentralized Hierarchical Access Control Scheme for the Medical Scenario. Proceedings of the Third Annual International Conference on Ubiquitous Access Control 2006, San Jose, CA, USA, pp. 1-6, IEEE Computer Society, 2006.

Related papers:

- Sigurd Eskeland. A Trust Framework and Authentication Protocol for Dynamic Assignment of Authorization in Medical Domains. Proceedings on Scandinavian Conference in Health Informatics, Arendal, Norway, pp. 50-56, 2004.

- Sigurd Eskeland. Efficient Hierarchical Conference Key Establishment in Wireless Networks. Proceedings on IASTED International Conference on Communication, Network and Information Security (CNIS 05), Phoenix, AZ, USA, pp. 94-98, ACTA Press, 2005.

- Sigurd Eskeland, Neeli Prasad. A Novel Hierarchical Team-Based Access Control Scheme for Wireless Networks. Proceedings of the 9th International Symposium on Wireless Personal Multimedia Communications (WPMC 2006), San Diego, CA, USA, September 2006.

- Sigurd Eskeland. Access Control by Secure Multi-Party EPR Decryption in the Medical Scenario. Proceedings on IASTED International Conference on Communication, Network and Information Security (CNIS 06), Boston, MA, USA, pp. 99-103, ACTA Press, 2006.

- Sigurd Eskeland, Vladimir Oleshchuk. Hierarchical Multi-Party Key Agreement for Wireless Networks. Proceedings on Third International Symposium on Information Assurance and Security (IAS 07), Manchester, England, pp. 39-43, 2007. IEEE Computer Society, 2007.

3.1 An overview of the results

Generic protocols	A, B, C, D, E
Protocols confined to the medical context	F, G, H, I

Table 1: Papers according protocol relevance

	Generic	Medical context
Ad hoc groups	A, B	F*, I
Long-term groups	C, D, E	F*, G, H

Table 2: Papers according to group composition and context

	Non-hierarchical	Hierarchical
Ad hoc groups	B, F*	A, I
Long-term groups	F*, D, E, H	C

Table 3: Papers according to group composition and hierarchy. The asterisk * denotes that temporal team property is irrelevant for this protocol.

Medical teams	H
Patients	F, G, I
Security administrators	G

Table 4: Papers according to granting entity

3.1.1 Considerations and overview

The included papers can be broadly categorized according to whether the included methods are generic, or apply specifically to the medical context (Table 1). Nevertheless, the generic methods are highly relevant for, but not limited to, the medical scenario. The presented methods are broadcast-oriented, and are therefore well-suitable for wireless networks.

Team aspects

The team (or group) aspect is essential in many of the methods of this thesis. We distinguish between groups according to whether they are ad hoc (short-term) and long-term. The latter means that the composition of such teams would be predefined for a long-term basis. Table 2 shows the papers according team lifetime (long-term/short-term) and context, i.e., generic, or specifically for the medical context.

The protocols presented in Papers A, B, I, support establishment of group-oriented cryptokeys for ad hoc groups. The key establishment is contributory, i.e., each of the pertaining participants contributes equally to the values of the established keys.

Long-term teams are characterized by a predefined composition. In this thesis, such teams are represented by long-term public keys. The protocols presented in Papers C, D, E, G, H, support long-term teams, whereof the cryptographic methods are based on threshold-oriented cryptography.

Hierarchical aspects

Both short-term and long-term teams can be hierarchical, meaning that medical teams would be composed of hierarchically ranked medical personnel like doctors and nurses. The medical information could be classified into sensitivity levels in

agreement with the user levels. The papers are categorized in Table 3 according to the hierarchical property.

Secure hierarchical group communication could be achieved by means of secure establishment of hierarchical session keys over insecure networks. A generic hierarchical key agreement protocol is presented in Paper A whereof an essential security property is that users of a given level can obtain data that pertains to their own and underlying security levels, while it must be computationally infeasible to obtain data pertaining to overlying security levels. Other hierarchical schemes are presented in Papers C and I.

Authorization of EPR access

The medical scenario is fast-changing and highly dynamic where medical practitioners provide care to new patients that are being continuously hospitalized. Since medical hospitals and institutions could house databases containing patient records for hundreds of thousands of people, access control can therefore not be based on long-term, predefined permission assignments between users and data objects. The presented methods in this thesis enable that such permission assignments can be established dynamically according to needs. The concept of roles, however, is well-suitable in context of user hierarchies, since a role could describe the hierarchical rankings of users. Accordingly, users could be granted access to data whose sensitivity level is in agreement with the user ranking, and to data of a lesser sensitivity level. It must accordingly be prohibited that participants may obtain access to data whose sensitivity level is above the privilege level of that participant.

In Chapter 2.4, we discussed that the need-to-know principle as a proper criterion dynamic establishment of EPR access. We also discussed the complementary issue of who to be in position of granting medical practitioners access to relevant medical data. The schemes presented in Papers F, G, H, I, put medical teams, patients and security administrators as granting authorities for establishment of EPR access.

Other considerations

Patient anonymity is addressed in the cryptographic scheme presented in Paper F. In this scheme, a patient can grant medical teams access to his or her EPR without revealing his or her identity. This scheme also provides secure data storage in the sense that the patient records are stored encrypted at the EPR server. Since there is no cryptokey tables, the consequent problem of secure key management is evaded due to that pertaining cryptokeys are confidentially established at the EPR server by means of the protected inputs of the patients and medical teams.

3.1.2 Contributions of the papers

Paper A: Hierarchical Multi-Party Key Establishment for Wireless Networks. In this paper, we present a generic hierarchical multi-party key agreement (conference key agreement) protocol that is well-suitable for wireless networks. A common property of practically all multi-party key agreement protocols is that they are non-hierarchical, i.e., they do not support user hierarchies. However, in the

medical context, medical practitioners are hierarchically ranked according to their job position. For example, medical doctors have a higher ranking than nurses, and it would be reasonable that doctors should access more privileged information than nurses. In contrast to existing schemes that provide secure deduction of hierarchically-arranged long-term keys, this protocol provides secure establishment of session keys in agreement with user hierarchy, allowing the users of each level to obtain the session key for their own and underlying levels. Acquisition of session keys of overlying levels is prohibited.

We also present a closely related hierarchical centralized key distribution protocol for totally-ordered and partially-ordered security classes.

Paper B: Public Group Key Cryptography. In this paper, we present an efficient conference key agreement scheme that enables ad hoc user groups to securely establish complementary session-based public/private key pairs. This concept is referred to in this paper as public group key cryptography. A corresponding multi-party signature provides certification of the public group key, cryptographically linking it to the originating the group participants. No online trusted third party is required for session key establishment.

In the medical scenario, secure communication for medical teams could be achieved by means of this group-oriented cryptosystem. By means of public group keys, EPR servers can securely transfer medical data to relevant teams of medical personnel.

Paper C: Efficient Hierarchical Group-Oriented Key Establishment and Decryption. In this paper, we present three related efficient generic cryptographic schemes for secure communication for hierarchically composed groups. The first one is a hierarchical key establishment scheme, but in contrast to the scheme in Paper A, it is not contributory. The scheme, like the one in Paper A, ensure that users can only obtain hierarchical session keys for their own and underlying levels, while it is prevented for overlying levels. The scheme is extended to a hierarchical public key cryptosystem based on the ElGamal cryptosystem, and furthermore to an ElGamal-based threshold decryption scheme.

The third scheme is a highly efficient broadcast-oriented threshold-decryption cryptosystem that requires only one round of broadcasting in the decryption phase. The threshold requirement enforce that at least t of n (where $t \leq n$) participants are required to collaborate in order to perform decryption.

These schemes are relevant for the medical scenario with regard to secure communication of medical data to medical teams of hierarchical composition.

Paper D: Collusion-resistant Threshold Decryption. In this paper, we propose a method applied to the threshold decryption scheme of Desmedt and Frankel [20] that prohibits colluding participants to deduce any of the secret coefficients of the underlying threshold Shamir secret sharing scheme [30].

Due to that medical care is in considerable degree provided by medical teams, secure group-oriented communication is essential in the medical context. As proposed in Paper G and H, threshold cryptography could be a proper security mechanism for the medical context. Most threshold-oriented cryptosystems incorporate the polynomial-based (t, n) threshold secret sharing scheme of Shamir, that is based on

Lagrange interpolation in order to reestablish the shared secret. However, Lagrange interpolation enable disclosure of the secret polynomial coefficients (including the secret shared key) given t user shares. Any participant holding the shared secret can subsequently carry out threshold-computations individually, thereby bypassing the threshold security requirement. This could be a serious problem when the threshold is low, for example 2 or 3. Disclosure of the polynomial coefficients enables establishment of new user shares. We refer to this as the collusion problem. In this paper, we propose a method for threshold decryption that prohibits computation of any of the secret coefficients of the polynomial of the underlying Shamir secret sharing scheme, and therefore solves the collusion problem in such cryptosystems.

Paper E: Collusion-Resistant Threshold Cryptosystems. This paper is similar to Paper D except that the method presented here requires less computations.

Paper F: Anonymity Preserving Authorization Granting In Medical Information Networks. Medical patient data often contains sensitive personal information. This could include sensitive information such as AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, genetic predispositions to diseases, drug addictions, etc. Due to this, it could in some cases be desirable that information identifying patients, e.g., names, addresses, personal identity numbers, etc., would not to be linked to electronic patient records. This paper addresses the need of hiding patient identities — in contrast to only keeping their medical data confidential. Another issue that is relevant for access control to electronic patient records, is what should be the criteria for EPR access to be granted. Patient consent is in general considered to be a reasonable basis for this.

In this paper, we present a scheme that enables consenting patients to anonymously grant medical teams authorization to access their EPRs, without revealing their true identities to the medical practitioners.

Paper G: EPR Access Authorization of Medical Teams Based on Patient Consent. In this paper, we present a cryptographic EPR access authorization scheme that incorporates patient consent as a basis for granting EPR access to medical teams or practitioners. This ensures that only the medical practitioners specified by a consenting patient are granted EPR access.

This scheme provides an increased level of data security because it assumes that all EPRs are stored encrypted at the EPR server. Each EPR is encrypted with its own unique and secret key that is unknown to all participants including the pertaining patient. The scheme provides secure and confidential establishment of EPR cryptokeys for subsequent decryption of the pertaining medical records. There are no cryptokey tables, but the cryptokey for a given EPR is temporarily restored at the EPR server for each session by means of the computations of the consenting patient in conjunction with the EPR server. The scheme is secure and prohibits deduction of the EPR cryptokeys, and that medical data to be disclosed without the collaboration of the consenting patient and a medical team.

A variation of the scheme allows an emergency or security team to grant EPR access on behalf of the patient if the patient is unconscious.

Paper H: Secure Team-Based EPR Access Acquisition in Wireless Networks. Medical teams providing care to a patient have a legitimate need to access the medical data of the concerning patient. The criterion for authorizing EPR access could be according to the consensus of a minimum number of concerned participants, like associated medical practitioners, e.g., a medical team. Group consensus could qualify as a basis for trust, and hence act as a proper basis for a medical team to acquire access to the required medical data.

In this paper, we present three closely related threshold-oriented cryptographic protocols providing secure team-based EPR access acquisition according to the consensus of a minimum number of associated medical participants. The schemes are broadcast-oriented, and are thus well-suitable for wireless networks. All schemes do also provide secure transfer of medical data.

Paper I: A Decentralized Hierarchical Access Control Scheme for the Medical Scenario. In this paper, we present a cryptographic access control scheme allowing patients to grant ad hoc medical teams authorizations to access their medical data. This scheme is based on the scheme in Paper A. The hierarchical aspects of teams are taken into account so that the modules of the patient record are to be accessed according to the individual privileges of the medical professionals of the team. Thus, more privileged users obtain larger portions of the data than less privileged users.

3.2 Summary of thesis contribution

The medical scenario is a complex and dynamic environment. Electronic information management and computer networks enable ubiquitous access possibilities to medical databases which may comprise hundreds of thousands of electronic patient records (EPRs). Such records may contain personal and highly sensitive patient data, and it is therefore necessary to limit the accessibility of such data to concerning medical personnel only. Since there would be a continuously flow of patients being hospitalized, access cannot be based on long-term, predefined permission assignments between medical practitioners and EPRs, but must rather be established dynamically according to needs.

In this thesis, we have identified legitimacy and the need-to-know principle as criteria for granting (establishment) of EPR access. This implies that only medical personnel going to provide medical care to a given patient should be granted access to the necessary medical data of the concerning patient they are going to provide care for. Proper enforcement of these principles would prohibit that non-legitimate personnel (medical or non-medical) would gain access to medical data without a legitimate reason. A relevant question would be what type of personnel should be in authority to grant legitimate personnel EPR access. We have considered consenting patients, security administrators (i.e., coalitions of such), and medical teams to be relevant, and in this thesis we have presented secure EPR access authorization schemes for each of these assumptions. Of these, there are two schemes based on patient consent, whereof one provides patient anonymity, and another provides secure data storage by long-term EPR database encryption without requiring cryptokey

tables.

The hierarchical ranking of medical practitioners is significant, since higher ranking implies more privileges. Medical practitioners of higher ranking (e.g., medical doctors) are accordingly in position to be entrusted access to a larger extent of sensitive information than practitioners of lower ranking (e.g., nurses). An essential hierarchical security requirement is to ensure that the medical practitioners are only granted access to EPR data whose confidentiality level is in agreement with the user ranking, and to EPR data associated with underlying confidentiality levels. Included are three security schemes complying with user hierarchy. In addition to the mentioned accessibility issues, protection of communicated data to and among multiple users is likewise of essential interest. A number of cryptographic methods are presented in this thesis that provide secure transmission of medical data over insecure wireless networks.

Due to the noted group-aspects, the proposed security schemes are mainly group-oriented. Although all apply for wireless medical networks, the schemes in Papers A-E are generic, and concern mainly secure group communication. Among others, these include secure establishment of hierarchical session keys by ad hoc groups, and secure establishment of public/private group keys by ad hoc groups. The schemes of the remaining Papers F-I apply specifically for the medical scenario.

[1] American Medical Association (AMA). Patient Confidentiality. See http://www.ama-assn.org/ama/pub/category/4610.html

[2] J. Bergmann et al. An e-consent-based shared EHR system architecture for integrated healthcare networks. International Journal of Medical Informatics, Vol. 76, Iss. 2-3, pp. 130-136, 2007.

[3] J. Biskup, G. Bleumer, Cryptographic protection of health information: cost and benefit, International Journal of Bio-Medical Computing 43, pp. 61-67, 1996.

[4] E. Coiera, R. Clarke. e-Consent: The design and implementation of consumer consent mechanisms in an electronic environment. Journal of the American Informatics Association, 11(2), pp. 129-140, 2004.

[5] P.A.B. Galpottage, A.C. Norris. Patient consent principles and guidelines for e-consent: a New Zealand perspective. Health Informatics Journal. Vol. 11, No. 1, pp. 5 – 18, SAGE Publications, 2005.

[6] T. Rindfleich. Privacy, information technology and health care. Communications of the ACM, Vol. 40, No. 8, 1997.

[7] For the Record: Protecting Electronic Health Information. Committee on Maintaining Privacy and Security in Health Care Applications of the National Information Infrastructure. National Academies Press, USA, 1997.

[8] G . Serour. Confidentiality, privacy and security of patients' health care information: FIGO Committee for the Ethical Aspects of Human Reproduction and Women's Health. International Journal of Gynecology & Obstetrics, Vol. 93, Iss. 2, pp. 189-190, 2006.

[9] R. A. Botha, J. H. P. Eloff. Separation of duties for access control enforcement in workflow environments. IBM Syst. J., Vol. 40, No. 3, pp. 666–682, 2001.

[10] R. Kruse. The Zipper, a method for using personal identifiers to link data while preserving confidentiality. Elsevier Science, 2001.

[11] Sweeney L. Guaranteeing Anonymity when Sharing Medical Data, the Datafly System. In Proceedings of the American Medical Informatics Association 1997 Annual Symposium. 1997.

[12] Amund Tveit et al. Anonymization of General Practioner's Patient Records. In Proceedings of the HelseIT'04 Conference, Trondheim, Norway, September 2004.

[13] X. Zou, B. Ramamurthy, S. Magliveras. Secure group communications over data networks. ISBN 0-387-22970-1. Springer Science-Business Media, Inc., 2005.

[14] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment. ISBN 3-540-43107-1, Springer-Verlag, 2003.

[15] A. Menezes et al. Handbook of Applied Cryptography. ISBN 0-8493-8523-7. CRC Press, 1997.

[16] B. Schneier. Applied Cryptography. ISBN 0-471-12845-7, Wiley, 1996.

[17] S. Rafaeli, D. Hutchinson. A survey of key management for secure group communication. ACM Computing Surveys, Vol.35, No.3, pp. 309–329, 2003.

[18] R.M. Needham, M. D. Schroeder. Using encryption for authentication in large networks of computers. Commun. ACM, 21(12), pp. 993-999, 1978.

[19] J.G. Steiner, B.C. Neuman, J. I. Schiller. Kerberos: An Authentication Service for Open Network Systems. In Usenix Conference Proceedings, Texas, 1988.

[20] Y.Desmedt, Y.Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307 – 315, Springer-Verlag, 1990.

[21] Y. Desmedt. Threshold cryptosystems. Advances in Cryptology, Proc. of Auscrypt'92, LNCS 718, pp. 3 – 14, Springer-Verlag, 1993.

[22] H. Ghodosi, S. Saeednia. A Modification to the Self-Certified Group-Oriented Cryptosystem Without a Combiner. Electronics Letters, vol. 37, no. 2, 2001.

[23] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings. Vol. 141, No. 5, pp. 307-313, 1994.

[24] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders, Eurocrypt 1994, pp. 194–204.

[25] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, vol. 547, pp. 522 – 526, Springer-Verlag, 1991.

[26] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner. Proc. of the 4th Australasian Conference on Information Security and Privacy. LNCS, vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[27] J. Pieprzyk, C. H. Li. Multiparty key agreement protocols. IEE Proceedings, Computer and Digital Techniques, Vol. 147, No. 4, pp. 229–236, 2000.

[28] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469–472, 1985.

[29] R. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Comm. of the ACM, Vol. 21, No. 2, pp. 120 – 126, 1978.

[30] A. Shamir. How to Share a Secret. Communications of the ACM, vol. 22, no. 11, pp. 612–613, 1979.

[31] W. Diffie, M. E. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, Vol. 22, No. 6, pp. 644- 654, Nov 1976.

Papers

Paper A

Hierarchical Multi-Party Key Establishment
for Wireless Networks

Sigurd Eskeland and Vladimir Oleshchuk

Hierarchical Multi-Party Key Establishment for Wireless Networks

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract

A common property of practically all multi-party key agreement protocols is that they are non-hierarchical, i.e., they do not support user hierarchies. However, in real life it is likely that members of groups and organizations differ in ranking according to their job positions, where participants of a lower ranking should be granted access to less privileged information than participants of higher rankings. Hierarchical access control (HAC) schemes address this issue, but in contrast to provide secure distribution of hierarchically-arranged short-term session keys, HAC schemes provide secure distribution of hierarchically-arranged long-term, predefined keys. In this paper, we present an efficient hierarchical multi-party key agreement protocol that is well-suitable for wireless networks. We also present a closely related hierarchical centralized key distribution protocol for totally-ordered and partially-ordered security classes.

Thus, it is reasonable that participants of a certain ranking could access more privileged information than participants of lower rankings.

1 Introduction

Key establishment protocols can be categorized into key transfer protocols and key agreement protocols. Regarding key transfer protocols, one entity generates and transfer a secret session key securely to one or more participants over an insecure network. In key agreement protocols, the users actively participate in the establishment of the secret shared session keys by collaboratively contributing themselves to the values of the keys. Key agreement protocols for groups or teams of more than two participants are known as multi-party key agreement or conference key agreement protocols. Several conference key agreement protocols has been previously proposed, see e.g., [1–3, 8], and common for these is that they are non-hierarchical, which of course is suitable when the group members have same ranking. However, in real life, people have different rankings according to their job positions, where individuals of lower ranking usually would be entrusted less confidential information than those of higher rankings.

For example, in the medical scenario, medical care is provided by medical teams that are composed of doctors and nurses where the doctors have a higher ranking than the nurses. Due to the sensitive nature of personal medical data, it could be required that medical practitioners of lower rankings should be privileged access to less sensitive or confidential medical information than those of higher rankings. It is also necessary that medical data is transferred securely to the legitimate team members. This could be achieved by means of encryption which requires secure establishment of shared secret keys according to the user hierarchy.

In this paper, we present a hierarchical multi-party key agreement protocol suitable for ad hoc wireless networks. It complies with an hierarchical arrangement of user classes (or security classes) where each user is associated with one security class that correspond to his or her job position. The protocol enables the participants to secretly establish one secret hierarchical session key for each security class – subsequently referred to as *class key*. Moreover, we present a closely related hierarchical centralized key distribution scheme, where class keys are originates from one party.

An essential security property is that the participants of any given security class can obtain the secret class keys that are established by their own and underlying security classes, while disclosure of class keys of overlying security classes is prevented.

2 Related work

The concept of hierarchical key agreement protocols seems to be somewhat absent in the literature. Hwang et al. [4] proposed a hierarchical key agreement protocol based on the multi-party key agreement key protocol in [7] that incorporates a hierarchy of classes that enables the participants to securely establish class keys for each class. Moreover, the participants of a given class can obtain the class keys of the underlying security classes, while it is prevented that class keys of overlying security classes can be disclosed. Unfortunately, the protocol provides no means to verify the user levels which allows any user to pretend to have a higher ranking than his or her legitimate ranking. Thus, the protocol fails to provide secure and trustable user hierarchies. Another major disadvantage is that it is highly inefficient due to that the number of rounds equals the number of participants. Eskeland proposed an efficient hierarchical key agreement protocol that requires only two rounds of broadcasting [5]. An improved version with increased computational efficiency and improved user authentication was published at the Third International Symposium on Information Assurance and Security [6] whereof this is an extended paper.

Hierarchical key establishment is not to be confused with hierarchical access control schemes (HAC) and tree-based key management schemes (TBKM). Hierarchical access control is a class of cryptographic schemes that supports deduction of long-term predefined cryptographic keys that are hierarchically arranged, so that users of a given security class are able to securely compute such keys associated with their own and underlying security classes, while computation of keys associated with overlying security classes is prevented.

While the hierarchical key establishment schemes presented in this paper provide secure ad-hoc establishment of a hierarchy of "fresh" sessions keys, (referred to as

class keys), HAC schemes do in contrast enable computation of predefined static keys from a key hierarchy. Providing computation of hierarchical predefined keys and not hierarchical sessions keys is reasonably a considerable limitation of the applicability and usefulness of such schemes.

However, due to access control purposes, many HAC schemes are compliant with user dynamics, i.e., inclusion and exclusion of users and corresponding renewal of hierarchical keys for the pertaining security classes. Examples of HAC schemes can be found in [14–17].

Tree-based key management schemes can be regarded as centralized key distribution where the users of a group establish a *key tree* where the users are arranged as leaf nodes of the tree. Due to the to tree structure, it allows them to obtain a common key that is the root. Thus, such schemes are not hierarchical due to that the users obtain one shared secret key. Examples are Tree-based Group Diffie-Hellman agreement [18], and others in [19, 20]. An essential issue about HAC and TBKM schemes is support of group dynamics (joining and leaving of users) so that protocol re-run is not necessary for each user update.

3 Preliminaries

3.1 Security requirements

The security requirements of the hierarchical multi-party key agreement protocol presented in Section 4 are as follows:

Security Requirement 1. *Class key confidentiality.* Only legitimate participants (or insiders) must be able to establish and obtain the class keys (or hierarchical session keys). It must not be possible to compute new class keys by means of former class keys of any security classes.

Security Requirement 2. *User key confidentiality.* It must be prevented that long-term secret user keys can be disclosed.

Security Requirement 3. *User authentication.* It must be securely established that each member is a genuine member of the claimed security class. Thus, user authentication must include certification of the pertaining security class of each user.

Security Requirement 4. *Forward secrecy.* Compromise of long-term secret user keys must not reveal formerly established class keys.

Security Requirement 5. *Onewayness.* The hierarchical multi-party protocol provides one secret class key for each level of the user hierarchy. It must be provided that each user of a given security class can only obtain class keys that are associated to his or her own and underlying security classes, and prevented that class keys of overlying classes can be disclosed.

The proposed schemes are broadcast-oriented and efficient, and are thus well-suitable for ad hoc wireless networks. Broadcasting efficiently distributes the key establishment messages from user user to the others, but allows an adversary to easily eavesdrop the key establishment messages. We can moreover assume that an adversary has been a former participant and may hold former keys. Consequently,

the security requirements must be satisfied in presence of passive adversaries with such capabilities.

An active adversary can modify (i.e., add, replace, replay) any broadcasted messages he or she wants. The adversary may, for example, attempt to impersonate any legitimate user by replaying old messages where the associated former class key is known. It must be infeasible to compromise the protocol without being detected or that the protocol does not terminate.

No online trusted party is required for computing class keys.

3.2 Hierarchical preliminaries

Let $\mathcal{U} = \{P_1, \ldots, P_m\}$ denote a team or group of m participants where each participant $P_i \in \mathcal{U}$ is associated with a hierarchy level and where L_i denotes the hierarchy level of P_i. We assume that each security level ℓ contains *one* security class $S_\ell \subseteq \mathcal{U}$ that includes all participants of that security level:

$$S_\ell = \{P_j \,|\, P_j \in \mathcal{U} \quad \text{and} \quad \ell = L_j\}$$

where $\ell \in \{1, \ldots, \lambda\}$ and λ denotes the top security level. We have that the security classes are partitions of $\mathcal{U}$ so that

$$\bigcup_{1 \le i \le \lambda} S_i = \mathcal{U} \qquad \text{and} \qquad S_i \cap S_j = \emptyset$$

where $i, j \in \{1, \ldots, \lambda\}$ and $i \neq j$. Thus, each participant $P_i \in \mathcal{U}$ is associated with *one* security class such that $P_i \in S_\ell$ for some $1 \le \ell \le \lambda$.

We denote the hierarchical ranking of the security classes according to the relation $\prec$ so that

$$S_i \prec S_j \quad \text{if} \quad i < j$$

which indicates that S_j has a higher ranking than S_i. The higher security level, the higher is the ranking in the hierarchy.

4 The protocol

In this section, we present the hierarchical multi-party key establishment protocol. It allows any composition of hierarchically ranked participants to establish of a corresponding hierarchy of conference keys over ad hoc wireless networks. Authentication allows any participant to detect if the hierarchical user arrangement is compromised, i.e., if a participant pretends to have a higher ranking than his or her legitimate ranking. The protocol is based on [5, 6] which are partly based on the multi-party key agreement protocol presented in [1].

4.1 User arrangement

In agreement with the directions in Section 3.2, the users are arranged in increasing order according to their ranking. Moreover, we assume that within each security class and across the security classes, the users are linearly ordered. This means that

the members of $\mathcal{U}$ form a sequence or string of users arranged in increasing order according to their ranking, where $P_1 \in S_1$ denotes the sequentially first user and $P_m \in S_\lambda$ denotes the sequentially last user. The users P_i and P_{i+1} are adjacent for $1 \leq i < m$.

The sequential user arrangement implies that if $P_i \in S_\ell$ is sequentially positioned first in S_ℓ, he or she is adjacent with $P_{i-1} \in S_{\ell-1}$ of the underlying class $S_{\ell-1}$. Thus, $\ell = L_{i-1}+1 = L_i$. Likewise, $P_i \in S_\ell$ is positioned at the end of S_ℓ if $\ell = L_{i+1}-1 = L_i$.

The participants could, for instance, be ordered within each security class by sorting according to their identities. To ensure different user order for each session, the users within each security class could be ordered according to $f(ID_i, T)$ where f is a hash function and T is a timestamp. Moreover, a change in the hierarchy, inclusion of new participants or participants leaving requires protocol re-run.

4.2 The protocol

In this subsection, we present the hierarchical key establishment scheme. It consists of an initialization stage, a key establishment stage with user authentication, and a key verification stage. The user authentication is analogous to the authentication scheme presented in [10, 11].

Initialization. A trusted authority (TA) is required to provide the long-term secret user keys which are the basis for the user authentication of the protocol. The TA has thus not a part in running the protocol.

According to the RSA cryptosystem [9], the TA selects two distinct secret large prime numbers p and q, and computes the composite modulus $n = p \cdot q$. The TA selects a public key e that is relative prime to $\phi(n) = (p-1) \cdot (q-1)$ and computes the secret key d so that $e \cdot d \equiv 1 \pmod{\phi(n)}$, where e and n are public. The TA selects a public element α that is of maximal order in $\mathbb{Z}_n^*$.

The TA computes for each user $P_i \in S_\ell$ an identifier as the hash of the concatenation of user identity ID_i and the pertaining user level L_i as $id_i = f(ID_i \,|\, L_i)$ where f is a secure one-way function. Based on id_i, the TA computes the secret user key

$$s_i = id_i^d \pmod{n}$$

The secret long-term user keys are confidentially distributed to the respective users.

Class key computation. The protocol goes as follows:

Step 1. Each participant $P_i \in \mathcal{U}$, $1 \leq i \leq m$, generates a random secret number $r_i \in \mathbb{Z}_n$, and computes and broadcasts

$$x_i = \alpha^{e \cdot r_i} \pmod{n}$$

Step 2. Each participant P_i for $2 \leq i \leq m-1$ computes

$$v_i = k_{i-1,i} - k_{i,i+1}^2 \pmod{n}$$

where $k_{i-1,i} = x_{i-1}^{r_i} \pmod n$ and $k_{i,i+1} = x_{i+1}^{r_i} \pmod n$ are secretly established Diffie-Hellman keys that are shared between P_i and P_{i-1}, and P_i and P_{i+1}, respectively. The squaring of the second term of v_i provides the one-way security property preventing $P_i \in S_\ell$ obtaining class keys K_l for higher classes where $l > \ell$.

The participants P_1 and P_m, who do not have *two* adjacent users, compute a number linking to the current session, say, $v_j = c$ for $j \in \{1, m\}$ where $c = f(x_1|x_2|\ldots|x_m)$ represents the current session due to the concatenation of the session dependent numbers and f denotes a secure one-way function. For authentication, each user $P_i \in \mathcal{U}$ computes

$$w_i = s_i \cdot \alpha^{r_i \cdot f(x_i, v_i, c)} \pmod n$$

Each participant $P_i \in \mathcal{U}$ broadcasts (ID_j, L_j, v_i, w_i).

Step 3. Authentication. Each participant $P_i \in \mathcal{U}$ authenticates the other participants $P_j \in \mathcal{U}$, $i \neq j$, by verifying

$$w_j^e \stackrel{?}{\equiv} id_j \cdot x_j^{f(x_j, v_j, c)} \pmod n$$

where $id_j = f(ID_j \,|\, L_j)$.

Step 4. Class key establishment. We define the class key K_ℓ for a given class S_ℓ as the DH key $k_{j,j+1}$ established by the sequentially first participant $P_j \in S_\ell$ of that class and the adjacent participant P_{j+1}. Thus, $\ell = L_j = L_{j-1} + 1$ where $L_0 = 0$ is the initial condition.

Due to that the users are sequentially arranged in agreement to the increasing ordering of the security classes, each participant is able to deduce the DH-keys of the preceding participants, i.e., participants of underlying security classes. The converse is prevented according to the one-way security property.

Each participant $P_i \in S_\ell$ computes the secret DH keys preceding participants $k_{j-1,j}$, $i > j$, according to the recurrence relation

$$k_{j-1,j} = v_j + k_{j,j+1}^2 \pmod n$$

Accordingly, $P_i \in S_\ell$ can compute the class key of his or her security class and underlying class keys K_γ, $1 \leq \gamma \leq \ell$, according to

$$K_\gamma = k_{j,j+1} = \alpha^{e r_j r_{j+1}} \pmod n$$

where P_j is the sequentially first participant of S_γ, i.e., $\gamma = L_{j-1} + 1$ where $L_0 = 0$.

Key verification. Each participant $P_i \in \mathcal{U}$ can verify that preceding participants $P_j \in \mathcal{U}$, $j < i$, have computed v_j according to the protocol. This is done by checking that

$$v_j^2 \stackrel{?}{\equiv} \widehat{k}_{j-1,j}^2 - 2 \cdot \widehat{k}_{j-1,j} \cdot k_{j,j+1}^2 + k_{j,j+1}^4 \pmod n$$

holds where $\widehat{k}_{j-1,j}$ is the deduced candidate DH key.

4.3 Example

To illustrate the protocol and hierarchical user arrangements, here is an example of 7 members of two security classes, $S_1 = \{P_1, P_2, P_3\}$ and $S_2 = \{P_4, P_5, P_6, P_7\}$, where $S_1 \prec S_2$. In agreement with Step 4, $P_7 \in S_2$, holding $k_{6,7}$, computes the class key of S_2 according to

$$K_2 = v_5 + (v_6 + k_{6,7}^2)^2 = k_{4,5} = \alpha^{er_4r_5} \pmod{n}$$

Next, $P_7 \in S_2$ computes the class key of the underlying security class S_1 according to

$$K_1 = v_2 + \left(v_3 + (v_4 + k_{4,5}^2)^2\right)^2 = k_{1,2} = \alpha^{er_1r_2} \pmod{n}$$

5 Security analysis

In this section, we show that the scheme is secure in agreement with the security requirements presented in Section 3.1.

Security Requirement 1. Class key confidentiality. The secrecy of class keys is based on the Diffie-Hellman computational problem, meaning that knowing $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$ where p is a large prime, it is computationally infeasible to find $\alpha^{x \cdot y} \pmod{p}$. Also note that due to the Discrete Logarithm Problem, it is computationally infeasible to deduce x given $\alpha^x \pmod{p}$. Accordingly, this holds for our scheme given where two participants P_i and P_{i-1} respectively hold the secrets r_i and r_{i-1}. Given the public numbers $\alpha^{e \cdot r_i} \pmod{p}$ and $\alpha^{e \cdot r_{i-1}} \pmod{p}$, the shared secret $\alpha^{e \cdot r_{i-1} \cdot r_i} \pmod{p}$ is protected due to the Diffie-Hellman computational problem.

Security Requirement 2. User key confidentiality. Note that w_i is composed of two secret factors, s_i and $\alpha^{r_i \cdot f(x_i, v_i, c)}$. Due to the RSA assumption where $\phi(n)$ is unknown because of the unknown factorization of n; given e, it is computationally infeasible to compute $e^{-1} = d \pmod{\phi(n)}$. This effectively prevents that secret user keys can be disclosed as $id_i^{(e^{-1})} \pmod{n}$ given id_i.

Accordingly, given x_i, it is computationally infeasible to obtain α^{r_i} since it is computationally infeasible to find $f(x_i, v_i, c)^{-1} \pmod{\phi(n)}$ due to the RSA assumption. Moreover, due to the Discrete Logarithm Problem, it is computationally infeasible to obtain r_i given x_i, which prevents establishment of α^{r_i}. This prevents computation of the secret factor $\alpha^{r_i \cdot f(x_i, v_i, c)}$ which accordingly protects the secret user key s_i.

Security Requirement 3. User authentication. User authentication is analogous to that in [10, 11], and is achieved due to the user signature computed by each user $P_i \in \mathcal{U}$. The user signature is constituted by

$$\begin{aligned} x_i &= \alpha^{e \cdot r_i} \pmod{n} \\ w_i &= s_i \cdot \alpha^{r_i \cdot f(x_i, v_i, c)} \pmod{n} \end{aligned}$$

where r_i is secretly known only by $P_i \in \mathcal{U}$, and s_i is the long-term secret user key.

An adversary may attempt to forge a valid signature w_i by raising it to a power $a'_j = f(x'_j, v'_j, c')$ that represents another context according to

$$w'_j = w_i^{a'_j} \pmod{n}$$

This will fail since w'_j will correspond to $id'_j = id_i^{a'_j} \pmod{n}$ which means that the adversary must overcome the difficulty of reversing the hash function f by finding ID'_j and L'_j so that $id'_j = f(ID'_j \mid L'_j)$.

A similar authentication scheme is found in [12] where the verification is analogous to

$$y_j^e \stackrel{?}{\equiv} ID_j \cdot x_j^{f(T)} \pmod{n}$$

where $x_j = \alpha^{e \cdot r_j} \pmod{n}$, $y_j = s_i \cdot \alpha^{r_j \cdot f(T)} \pmod{n}$ and T is a timestamp. This scheme is not resistant to the Extended Euclidian Algorithm attack [13]. If e and $f(T)$ are relatively prime, we can find two integers u, v, so that $e \cdot u = 1 + f(T) \cdot v$. An adversary can thus pick a valid id_j, and compute $x_j = ID_j^v \pmod{n}$ and $y_j = ID_j^u \pmod{n}$. The attack succeeds in the scheme in [12] because

$$y_j^e = ID_j \cdot x_j^{f(T)} = (ID_j^u)^e = ID_j \cdot (ID_j^v)^{f(T)} \pmod{n}$$

This attack is thwarted in our scheme since x_j is included in certifying power $f(x_j, v_j, c)$.

Security Requirement 4. Forward secrecy is defined as when a long-term key is compromised, class keys that were previously established using that long-term key should not be compromised too [8, p. 50]. Compromise of long-term user keys would enable an adversary to obtain $\alpha^{r_i \cdot f(x_i, v_i, c)}$ given w_i, $1 \leq i \leq m$. As noted, the secret r_i is protected due to the Discrete Logarithm Problem. Thus, the pertaining $k_{i,i+1}$ or $k_{i-1,i}$ cannot be deduced by means of long-term user keys, and forward secrecy is provided.

Security Requirement 5. Onewayness. The one-way property prevents that $P_i \in S_\ell$ can compute K_l if $S_\ell \prec S_l$. This is ensured by squaring the last term of v_j. Since n is the product of two large secret primes, the value of $\phi(n)$ is unknown, and it is thus computationally infeasible to find roots in $\mathbb{Z}_n$. In order to obtain the succeeding secret DH key $k_{i+1,i+2}$, $P_i \in \mathcal{U}$, holding $k_{i,i+1}$, must solve

$$\begin{aligned} k_{i+1,i+2} &= \sqrt{k_{i+1,i+2} - v_{i+1}} \pmod{n} \\ &= \sqrt{k_{i,i+1} - (k_{i,i+1} - k_{i+1,i+2}^2)} \pmod{n} \\ &= \sqrt{k_{i+1,i+2}^2} \pmod{n} \end{aligned}$$

which is computationally infeasible since the factorization of n is unknown.

It is essential that each user $P_i \in \mathcal{U}$ computes v_i according to the protocol. The protocol could be subverted if a malicious user $P_i \in \mathcal{U}$ would broadcast $v_j = k_{j-1,j} - k_{j,j+1} \pmod{n}$ since this would break the onewayness security property, allowing P_{i-1} to deduce $k_{i,i+1}$. Attempts of such violations will, however, be detected by an

honest participant who will deduce the candidate DH key as $\widehat{k}_{j-1,j} = v_j + k_{j,j+1}^2 = k_{j-1,j} - k_{j,j+1} + k_{j,j+1}^2$. This will cause that the verification

$$v_j^2 = (k_{j-1,j} - k_{j,j+1})^2 \neq \widehat{k}_{j-1,j}^2 - 2\,\widehat{k}_{j-1,j}\,k_{j,j+1} + k_{j,j+1}^4 \pmod{n}$$

does not hold, and the protocol aborts.

6 Generalizing the protocol

The protocol presented in the previous section can be put in a more general form. In the general representation, any two-party key establishment scheme, user authentication scheme and one-way function could be used as building blocks, whereas the given security assumptions would consequently then rely on the actual security properties of the applied building blocks. Since user authentication may or may not be integrated to key establishment, we will here only consider the general class key establishment.

The same assumptions about user alignment apply for the generalized protocol so that the users are sequentially arranged in increasing order according to their ranking. Given any secure two-party key establishment protocol, each participant $P_i \in \mathcal{U}$ establishes the secret session keys $k_{i-1,i}$ (if $i > 1$) and $k_{i,i+1}$ (if $i < m$) shared respectively with $P_{i-1} \in \mathcal{U}$ and $P_{i+1} \in \mathcal{U}$.

Then $P_i \in \mathcal{U}$ computes and broadcasts

$$v_i = k_{i-1,i} - f(k_{i,i+1})$$

where f is a secure one-way function. Note that the computations could be modular or non-modular. This could in practice depend on the integer size of actual implementations. However, avoiding the modulus operator would obviously increase the computational efficiency.

A further generalization pertains operators which has so far been confined to subtraction and addition. Division (and subsequent multiplication for class key restoration) would work fine, but should be modular since computations involving real numbers should be avoided:

$$v_i = \frac{k_{i-1,i}}{f(k_{i,i+1})} \pmod{p}$$

Nevertheless, addition is more efficient than multiplication and would thus be more preferable. Lastly, the bitwise XOR operator could be applied for the best computational efficiency:

$$v_i = k_{i-1,i} \oplus f(k_{i,i+1})$$

Application of the bitwise XOR operator requires consequently that the number of bytes of the keys $k_{i-1,i}$ and the output of f are equal.

Computation of preceding keys and class keys would be in agreement with the respective recurrence relations

$$k_{j-1,j} = \begin{cases} v_j + f(k_{j,j+1}) \\ v_j \cdot f(k_{j,j+1}) \pmod{p} \\ v_j \oplus f(k_{j,j+1}) \end{cases}$$

where, in agreement with Step 4 of the scheme presented in Section 4, $P_i \in S_\ell$ can compute $K_\gamma = k_{j,j+1}$ for $1 \leq \gamma \leq \ell$ if P_j is sequentially first in S_γ, i.e., $\gamma = L_{j-1} + 1$ where $L_0 = 0$.

Note that, for example, if the applied one-way function is a hash-function, the key verification step in Section 4 would not work.

7 Centralized key distribution

In this section, we present a centralized key distribution scheme that is based on the key establishment protocol in the previous section. By centralized we mean that one entity initiates the protocol by means of hierarchical public parameters of the pertaining group. This is in contrast to the previous protocol where all participants depend on the others in order to establish and deduce the secret hierarchical session keys.

As follows, the centralized key distribution scheme is presented respectively for both totally-ordered and partially-ordered security classes. Moreover, all participants of each security class share a long-term secret key that is associated to that class.

7.1 Security properties

In the previous protocol, the hierarchical session keys (or class keys) are established and deduced as a function of user inputs where each user contributes with session-dependant inputs. In the following centralized protocols, the class keys are established as a function of the public parameters representing a team, the secret long-term hierarchical keys and a random number. Thus, the correct class key can only be established by means of the proper secret long-term hierarchical key.

An essential security property of our scheme is that although participants of a given security class may compute class keys of underlying security classes, it should be prevented that any given class may deduce secret *long-term hierarchical keys* of other classes, i.e., long-term hierarchical key confidentiality. A reason why this is essential, is that such keys could be used for other applications and purposes as well whereof participants of other security classes may not be involved with. Consequently, it must be ensured that long-term hierarchical keys remain undisclosed, even for participants of higher rankings.

Note that since any party could initiate key distribution without not necessarily possessing any pertaining long-term secret user keys, this party would be prevented from obtaining the corresponding hierarchical session keys. Since key transfer protocols allow one party to securely transfer a secret key to other parties, the following protocol qualifies as a key transfer protocol only if the initiating party possesses the pertaining long-term secret user keys.

The centralized protocols do not have explicit user authentication like the previous protocol. However, since user authentication is based on the assumption that only the legitimate users hold the pertaining secret long-term keys (whereof key correctness depends), user authentication is an implicit property of the protocol. Thus, the centralized protocols provide implicit user authentication, key secrecy

and hierarchical one-way security property. The security properties comply with the previous protocol except that forward secrecy is not supported.

7.2 Totally-ordered centralized protocol

In this subsection, we present the totally-ordered centralized version of the protocol.

Initialization. Let $n = p \cdot q$ where p and q are two large distinct primes, and let α be an element of maximal order in $\mathbb{Z}_n^*$. The trusted authority (TA) that sets up the scheme randomly generates λ secret long-term hierarchical user keys, $k_j \in \mathbb{Z}_{\phi(n)}$, $j \in \{1, \ldots, \lambda\}$, so that each user in the security class S_j is confidentially handed the pertaining k_j. The TA computes for each $S_j \subseteq \mathcal{U}$ the public parameters

$$Y = \{y_j = \alpha^{k_j - 2 \cdot k_{j+1}} \pmod{n} \mid 1 \leq j < \lambda\}$$

Key establishment. One particular party is required to initiate the protocol. We refer to this party as the registry. This could be any of the participants or an arbitrary outsider. The protocol goes through the following steps:

Step 1. The registry selects a random number r, computes and broadcasts

$$z_j = y_j^r \pmod{n} \quad \text{and} \quad R = \alpha^r \pmod{n}$$

for each security class $S_j \subseteq \mathcal{U}$.

Step 2. Each participant in S_ℓ computes the class key referring to his or her security class according to

$$K_\ell = R^{k_\ell} \pmod{n}$$

In general, each participant in S_ℓ computes the class keys K_i of his own and the underlying security classes S_i, $1 \leq i \leq \ell$ according to

$$\begin{aligned} K_i = R^{k_i} &= z_i \cdot z_{i+1}^2 \cdots z_{j-1}^{(2^{\ell-1})} \cdot R^{(2^\ell \cdot k_\ell)} \pmod{n} \\ &= R^{(2^\ell \cdot k_\ell)} \cdot \prod_{j=0}^{\ell-1} z_{i+j}^{(2^j)} \pmod{n} \end{aligned}$$

Example. Given a 4 level totally-ordered hierarchy, each participant in S_4 would compute $K_1 = z_1 \cdot z_2^2 \cdot z_3^4 \cdot R^{8 \cdot k_4} \pmod{n}$, etc.

7.3 Partially-ordered centralized protocol

The protocol for partially-ordered security classes is basically the same as the totally-ordered centralized protocol other from its more general hierarchical capabilities.

Definitions and notation. Let $\mathcal{U} = \{S_1, \ldots, S_\lambda\}$ be λ disjoint security classes. Let $H \subseteq \mathcal{U} \times \mathcal{U}$ be the binary relation where $(S_i, S_j) \in H$ *iff* S_i is an immediate predecessor of S_j and where the users in S_i have a higher security clearance than the users in S_j. A partially ordered set $(\mathcal{U}, H)$ can be represented by a Hasse diagram where an edge from S_i to S_j represents $(S_i, S_j) \in H$.

Let H^* denote a reflexive transitive closure of H. Let $S_i \preceq S_j$ *iff* $(S_i, S_j) \in H^*$. This means that the users in S_i have a security clearance higher than or equal to the users in S_j. Fig. 1 shows an example of a partially ordered hierarchy defined by the relation $H = \{(S_6, S_5), (S_6, S_4), (S_4, S_3), (S_4, S_2), (S_3, S_1), (S_2, S_1)\}$. For example, $(S_6, S_4), (S_4, S_2) \in H$ implies that $(S_6, S_2) \in H^*$.

Initialization. The TA generates long-term secret user keys as in the previous subsection, and computes the long-term public hierarchical parameters Y according partially-ordered hierarchical structure of the group $\mathcal{U}$.

For each edge $(S_i, S_j) \in H$, the TA computes the public parameters

$$Y = \{y_{i,j} = \alpha^{k_i - 2 \cdot k_j} \pmod{n} \mid (i,j) \in I_H\}$$

where $I_H = \big\{(i,j) \,|\, (S_i, S_j) \in H\big\}$. Thus, $|Y| = |H|$. The example in Fig. 1 corresponds to $Y = \big\{y_{6,5},\, y_{6,4},\, y_{4,3},\, y_{4,2},\, y_{3,1},\, y_{2,1}\big\}$.

Key establishment. The registry selects a random number r, computes and broadcasts

$$Z = \{z_{i,j} = y_{i,j}^r \pmod{n} \mid y_{i,j} \in Y\} \quad \text{and} \quad R = \alpha^r \pmod{n}$$

Next, each participant in S_ℓ computes the conference key referring to his or her security class according to

$$K_\ell = R^{k_\ell} \pmod{n}$$

If there exists an edge $(S_i, S_j) \in H$, then K_j can be computed if K_i is known:

$$K_j = z_{i,j} \cdot K_i^2 \pmod{n} \quad \text{where} \quad z_{i,j} \in Z$$

Thus, in general, if $S_{\ell,j} \in H^*$, each participant in S_ℓ can recursively compute K_j.

8 Conclusion

In this paper, we have presented an efficient hierarchical multi-party key agreement protocol that enables an arbitrary number of users of λ security classes to securely compute a secret class key for each security class. It provides user authentication, and allows users in a given security class to compute the secret class keys of the same and underlying security classes, while it is prevented that any user can obtain class keys of overlying security classes. The scheme is broadcast-oriented and requires only two rounds of broadcasting, and is thus well-suitable for wireless networks.

We have moreover presented a centralized hierarchical key distribution scheme based on the former that supports totally-ordered and partially-ordered user hierarchies.

References

[1] M. Burmester, Y. Desmedt. A secure and efficient conference key distribution system. In proc. of Eurocrypt'94, LNCS, vol. 950, pp. 275 – 286, Springer-Verlag, 1994.

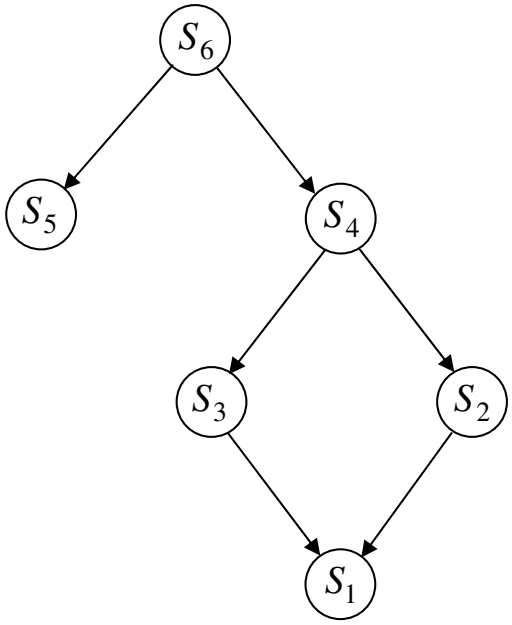

Figure 1: Example of a partially ordered hierarchy

.

[2] M. Just, S. Vaudenay. Authenticated multi-party key agreement. In proc. of Asiacrypt'96, LNCS, vol. 1163, pp. 36 – 49, Springer-Verlag, 1996.

[3] S. Saeednia, R. Safavi-Naini. Efficient identity-based conference key distribution protocols. ACISP'98, LNCS 1438, pp. 320–221, Springer-Verlag, 1998.

[4] M. Hwang, W. Tzeng. A conference key distribution scheme in a totally-ordered hierarchy. ICOIN 2003, LNCS 2662, pp. 757 – 761, Springer-Verlag, 2003.

[5] S. Eskeland. Efficient Hierarchical Conference Key Establishment in Wireless Networks. IASTED International Conference on Communication, Network and Information Security '05, pp. 94 – 98, Acta Press, 2005.

[6] S. Eskeland, V. Oleshchuk. Hierarchical Multi-Party Key Agreement for Wireless Networks. Third International Symposium on Information Assurance and Security '07, pp. 39 – 43, IEEE Computer Society, 2007.

[7] K. Koyama, K. Otha. Identity-based conference key distribution system. Advances in Cryptology, Crypto'87, pp. 194 – 202, Springer-Verlag, 1987.

[8] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment. ISBN 3-540-43107-1, Springer-Verlag, 2003.

[9] R. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Comm. of the ACM, Vol. 21, No. 2, pp. 120 – 126, 1978.

[10] K. Koyama, K. Ohta. Security of improved identity-based conference key distribution systems. Adv. in Cryptology - EuroCrypt '88, LNCS 330, pp. 11 – 19, Springer-Verlag, 1988.

[11] K. Koyoma. Secure conference key distribution schemes for conspiracy attack. Advances in Cryptology, Crypto '92, pp. 449–453, Springer-Verlag, 1992.

[12] W. Yang, S. Shieh. Password authentication schemes with smart cards. Computer & Security, vol. 18, no. 8, pp. 727 – 733, 1999.

[13] K. Chen, S. Zhong. Attacks on the (enhanced) Yang-Shieh authentication. Computer & Security, vol 22, no. 8, pp. 725 – 727, 2003.

[14] F. Kuo, V. Shen, T. Chen, F. Lai. Cryptographic key assignment scheme for dynamic access control in a user hierarchy. IEE Proc. Computers & Digital Techniques, Vol 146, No. 5, 1999, pp. 235 – 240.

[15] C. Lin. Dynamic key management schemes for access control in a hierarchy. Computer communications, Vol. 20, No. 15, pp. 1381 – 1385, 1997.

[16] C. Chang, C. Lin, W. Lee, P. Hwang. Secret sharing with access structures in a hierarchy. AINA, Vol. 2, pp. 31 – 34, 2004.

[17] X. Zou, B. Ramamurthy, S. Magliveras. Chinese Remainder Theorem based hierarchical access control for secure group communications. ICICS, LNCS, Vol. 2229, pp. 381 – 385, 2001.

[18] Y. Kim, A. Perrig, G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. Proc. of 7th ACM CCS, pp. 235 – 244, 2000.

[19] L. Dondeti, S. Mukherjee, A. Samal. DISEC: a distributed framework for scalable secure many-to-many communication. Proc. of 5th IEEE ISCC, pp. 693 – 698, 2000.

[20] A. Sherman, D. McGrew. Key establishment in large dynamic groups using one-way fucntion tree. IEEE transactions on Software Engineering, Vol. 29, No. 5, pp. 444 – 458, 2003.

Paper B

Public Group Key Cryptography

Sigurd Eskeland

Public Group Key Cryptography

Sigurd Eskeland
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
sigurd.eskeland@uia.no

Abstract

Several conference key establishment protocols have in the recent years been proposed for the purpose of secure group communication. In this paper, we introduce the concept of public group key cryptography as an extension to regular conference key agreement, and propose an efficient and practical public group key cryptosystem that enables participants of groups, without an online trusted third party, to flexibly establish public/private key pairs that represent the groups. By means of private group keys, secure communication is provided internally between the group members, and moreover, outsiders can by means of the public group key address the group confidentially. A multi-party signature provides certification of the public group key towards the group participants.

1 Introduction

With the evolution of Internet and wireless networks and with increasing focus on group collaboration, there has been a correspondingly increased focus on secure group communication, i.e., two or more participants that are physically located on different locations can confidentially establish on a secret key over an insecure public network. Key establishment protocols can be categorized into two concepts: 1) *Key agreement*, where each participant contributes equally to the value of the session key, and 2) *key distribution* (or key transfer), where a trusted party or participant generates a secret session key and transfers it confidentially and securely to the legitimate participants. Key agreement protocols involving more than two participants are called conference key agreement or multi-party key agreement protocols.

There may be situations when it is desirable that outsiders can address a group securely and confidentially. A naive approach for this in the context of secure group key establishment protocols could be to include the outsider into the group, and then re-run the protocol in order to re-establish a new secret group key. The confidentiality of the secret information shared within the group would accordingly be compromised, since the new participant would be sharing the key with

the group and would therefore have access to subsequently exchanged confidential information that is only to be disclosed to the original members.

A proper solution could be to incorporate a public key that would publicly represent the group where the complementary secret key would consequently be known to the group members only. This way, the following is achieved: 1) The group members can now communicate securely among themselves, and 2) outsiders can address the group securely by means of the public key.

Centralized solutions involving online trusted third parties (TTP) providing key generation and secure key distribution, may conform poorly to distributed and wireless environments. By extending the concept of conference key agreement, we propose a public group key cryptosystem where group members can collaboratively establish corresponding public/private group keys pairs over a wireless network without involving an online TTP. In order to make the public keys trustable, some kind of certification is required. The proposed cryptosystem provides identity-based signatures that certify public group keys toward identities of the originating users and time of establishment.

1.1 Related work

Threshold-oriented cryptosystems is class of group-oriented cryptosystems that are based on the concept of public key cryptography where a group, consisting of a number of members, is represented by a public key. See see [1–5] for examples of such schemes. Although our scheme provides a public key representation of groups, it differs fundamentally from those due to that it is not a threshold scheme, and that it is identity-based. We will nevertheless discuss a few such schemes since they partially relates to our scheme.

The first threshold-oriented cryptographic protocol was proposed in [2]. In this scheme, there is one public key and a number of corresponding secret shares that together represent the corresponding secret key. The public key represents the group, and the public key and secret shares are generated by a trusted third party (TTP). Each participant is handed a share by which each participant computes a partial decryption of the ciphertext. A minimum number of participants, defined by the threshold, must combine their partial decryptions in cooperation with a trusted party to reconstruct the plaintext. Due to the (unsigned) public key, the scheme provides anonymity, keeping the members of the group anonymous to outsiders.

In [4,5], a group-oriented cryptosystem is presented. In this system, the outsider that wants to address the group, encrypts the message with each of the public keys of the participants of the group, and sends each of these cryptograms to the corresponding participants. In order to decrypt, a minimum number of the participants, the threshold, must cooperate to restore the plaintext, without the help of a trusted combiner function.

The protocol presented in [4] and subsequently updated in [5] is an identity-based threshold-decryption scheme based on self-certifying public keys [13]. The partic-

ipants compute the plaintext themselves from their partial decryptions, thereby eliminating an online trusted party for this. However, this scheme requires the sender to encrypt the plaintext for each of the members of the corresponding group that he or she wants to confidentially address. In contrast to encrypting once by means of one public key, this scheme imposes an unreasonably high overhead considering computation and bandwidth.

2 Public group key

By means of a public group key, outsiders can confidentially address groups and the group members can communicate secretly among themselves. The following list indicates security requirements for the proposed key establishment protocol:

- The protocol must resist passive attacks (i.e., eavesdropping) so that secret group keys remain confidential to the concerning parties only.
- The protocol must resist active attacks so that attempts of substitution of and alternation of key establishment messages will be detected.
- The establishment of group keys is to be performed in a distributed and contributory fashion so that each participant contributes equally to the key, thereby providing unique and "fresh" keys for each session.
- Forward secrecy is defined as if a long-term user key is compromised, session keys previously established with that long-term user key cannot be compromised [12, p. 50]. The protocol must provide forward secrecy.
- Public group keys must be certifiable offline (without an online trusted party) towards the identities of all the originating participants, and time of establishment.
- No online TTP or combiner functions.
- A nice but not really mandatory property is public reconstruction of the public group key, making the public group key privately verifiable towards the secret group key, thereby providing implicit key authentication.

An important property for a public key is to provide proof of context. In order for a public key to be *trustworthy* for the outside world or valid, outsiders must be certain that the owner of this public key is *really* the person he thinks it is. Otherwise, an unauthorized individual could cheat and obtain this confidential information of substituting public keys with his own. It is therefore essential that the association between public keys and their owners can be verified. In the context of our scheme, individuals running the protocol would consequently constitute and thereby define the groups that the established keys would represent. It is highly essential that the association between group-established public group keys and the identities of the originating group members is certifiable, including the time of establishment. The issue of statically linking public keys with identities is addressed by certificates and in [13] by the concept of self-certified public keys. This does not comply with group key agreement where keys are established

session-wise by a number of participants, without a TTP. The proposed protocol employs a homomorphic identity-based signature scheme for establishment of multi-party signatures verifying the corresponding public group keys.

3 A practical public group key cryptosystem

In this section, the public group key protocol is presented. In 1994, Burmester and Desmedt generalized the well-known Diffie-Hellman two-party key agreement protocol into a conference key agreement protocol [6]. In [7], this was moreover generalized into a multiplicative and an additive variant having the secret keys on the form respectively as $K = k_{1,2} \cdot k_{2,3} \cdot \ldots \cdot k_{m,1}$ and $K = k_{1,2} + k_{2,3} + \ldots + k_{m,1}$. Our protocol is based on the additive variant and is very efficient as it requires only two rounds (two broadcasts per participant), and a third round for establishment of the multi-party public group key signature. Due to broadcasting, it is well suitable for wireless environments. The protocol is identity-based and each participant is assigned identity-based long-term secret user keys.

3.1 Setup

A trusted center, a trusted third party (TTP) computes the product of two secret, large prime numbers $n = p \cdot q$. According to the RSA cryptosystem, the center selects a number e that is relative prime to $\phi(n)$, and computes a secret integer d so that $e \cdot d \equiv 1 \pmod{\phi(n)}$. The TTP publishes n and e.

The only purpose for the TTP is to provide long-term secret ID-based user keys. Long-term user keys are required for user computation of signatures certifying key agreement messages and public group keys. The TTP computes for each user P_i the long-term secret user keys, $s_i = ID_i^d \pmod{n}$ and $h_i = \alpha^{d \cdot f(ID_i)} \pmod{n}$, where f denotes a secure hash-function, d is only known by TTP, and α is a public primitive root of $GF(\mathrm{p})$ and $GF(\mathrm{q})$.

It is assumed that the user identities consist of meaningful information like names, addresses, etc., or that the identities are posted in tamper-free tables or secure bulletin boards for verification. Otherwise, adversaries can produce fake user keys s' representing *meaningless* identities id' according to $id' = s'^e \pmod{n}$.

3.2 Group key computation

The group $\mathcal{U} = \{P_1, \ldots, P_m\}$, consisting of m members, is arranged according to a logical ring structure so that P_m and P_1 are an adjacent pair of users, and that $P_{m+i} = P_i$. Note that in general that the indexing is circular, so that if the index j is $j < 1$, then it corresponds to the index $i = j + m$, or if $j > m$, then the corresponding index is $i = j - m$, where $1 \leq i \leq m$. Thus, the indices j and $j + m$ are equivalent: $j \equiv j + m$. The protocol goes as follows:

Round 1. Each adjacent pair of users, P_i, P_{i+1}, establish a secret key by means of a secure two-party key agreement (TPKA) protocol. That is, every user P_i establishes with users P_{i+1} and P_{i-1} two secret keys, $k_{i,i+1}$ and $k_{i-1,i}$, respectively. An efficient TPKA protocol for broadcasting is to employ authenticated Diffie-Hellman key agreement.

Each user P_i

1. generates a secret number r_i, and computes
$$x_i = \alpha^{f(ID_i)\cdot r_i} \pmod n \quad \text{and} \quad y_i = s_i \cdot h_i^{r_i \cdot f(x_i, T_i)} \pmod n$$
where f is a secure one-way function, T_i is a time-stamp indicating the current time and x_i acts as a verifiable ephemeral public key of P_i.
2. broadcasts (ID_i, x_i, y_i, T_i).
3. verifies for each $P_j, j = 1, \ldots, m$, that
$$y_j^e \stackrel{?}{\equiv} ID_j \cdot x_j^{f(x_j, T_j)} \pmod n$$
and $|T' - T_j| < \Delta T$ where T' is the current time and ΔT is a margin due to time delay and clock inconsistencies.

Round 2. Each user P_i

1. establishes with P_{i-1} and P_{i+1} two secret keys, respectively
$$k_{i-1,i} = x_{i-1}^{r_i} \pmod n \quad \text{and} \quad k_{i,i+1} = x_{i+1}^{r_i} \pmod n$$
2. computes
$$\begin{aligned} v_i &= k_{i,i+1} - k_{i-1,i} \\ X_i &= \alpha^{f(ID_i)\cdot k_{i,i+1}} \pmod n \\ Y_i &= s_i \cdot h_i^{k_{i,i+1}\cdot f(X_i, v_i, c)} \pmod n \end{aligned}$$
where $c = f(x_1, x_2, \ldots, x_m)$. Note that c is linking the contributions of each participant of the first round to the second round.
3. broadcasts (ID_i, X_i, Y_i, v_i).
4. verifies whether
$$Y_j^e \stackrel{?}{\equiv} ID_j \cdot X_j^{f(X_j, v_j, c)} \pmod n$$
holds for $j = 1, \ldots, m$.
5. computes the common secret group key
$$\begin{aligned} K &= v_{i-1} + 2 \cdot v_{i-2} + \ldots + (m-1) \cdot v_{i+1} + m \cdot f(ID_i) \cdot k_{i,i+1} \\ &= m \cdot f(ID_i) \cdot k_{i,i+1} + \sum_{j=1}^{m-1} j \cdot v_{i-j} \end{aligned}$$
where $v_j = v_{m+j}$.

6. computes the public group key according to

$$\mathcal{P} = \prod_{j=1}^{m} X_j = \alpha^K \pmod{n}$$

By verifying that $\prod_{j=1}^{m} X_j \stackrel{?}{\equiv} \alpha^K \pmod{n}$, $\mathcal{P}$ and K is implicitly authenticated due to that each X_i, $i = 1, \ldots, m$, is a factor of the public group key.

3.3 Confirmation and certification

In the last round, the participants collaboratively compute a signature for the public group key $\mathcal{P}$. The signature provides public verification of $\mathcal{P}$, and thus accordingly confirms the secret group key for the participants.

Round 3. Each user P_i

1. computes and broadcasts an individual signature of public group key

$$\mathcal{S}_i = s_i \cdot h_i^{K \cdot f(\mathcal{P}, \bar{T})} \pmod{n}$$

   where $\overline{T}$ is the average of T_j, $j = 1, \ldots, m$, from the first round.
2. verifies the individual signatures for each P_j, $j = 1, \ldots, m$:

$$\mathcal{S}_j^e \stackrel{?}{\equiv} ID_j \cdot \mathcal{P}^{f(\mathcal{P}, \overline{T}) \cdot f(ID_j)} \pmod{n}$$

3. computes the multi-party signature $\mathcal{S} = \prod_{j=1}^{m} \mathcal{S}_j \pmod{n}$ that corresponds to

$$\mathcal{S} = \Big(\prod_{j=1}^{m} ID_j^d \Big) \cdot \alpha^{d \cdot K \cdot f(\mathcal{P}, \overline{T}) \cdot \sum_{j=1}^{m} f(ID_j)} \pmod{n}$$

Finally, the public group key and its certifying parameters is published: $(\mathcal{P}, \mathcal{S}, \overline{T}, id)$ where $id = \{ID_i \,|\, 1 \leq i \leq m\}$. $\mathcal{P}$ is certified according to

$$\mathcal{S}^e \stackrel{?}{\equiv} \Big(\prod_{i=1}^{m} ID_i \Big) \cdot \mathcal{P}^{\cdot f(\mathcal{P}, \overline{T}) \cdot \sum_{i=1}^{m} f(ID_i)} \pmod{n}$$

Since the group key signature will detect errors and inequalities that would be detected in step 2, step 2 can be omitted.

3.4 Employment

By employing the ElGamal cryptosystem, outsiders are now able to encrypt messages by means of $\mathcal{P}$ so that only the legitimate group members associated to $\mathcal{P}$ are able to individually decrypt. A plaintext message M is encrypted by $a = \alpha^r \bmod n \quad \text{and} \quad b = M \cdot \mathcal{P}^r \bmod n$ where r is a random number secretly chosen by the sender, and subsequently decrypted by $M = b \cdot a^{-K} \bmod n$. The group participants can correspondingly communicate secretly by $\mathcal{P}$, or by symmetric encryption by K.

4 Security analysis

In this section, we consider the security of the proposed cryptosystem according to the previously stated security requirements.

Security Requirement 1 *The protocol must provide secret group key confidentiality, and must thus be secure against passive attacks.*

Proof. The secret values $k_{i,i+1}$ are established due to authenticated Diffie-Hellman-based key establishment where secrecy is achieved due to the Computational Diffie-Hellman Problem. That is, knowledge of $\alpha^x \bmod p$ and $\alpha^y \bmod p$ does not provide $\alpha^{xy} \bmod p$. After the first round, $k_{i,i+1}$ is known only by P_i and P_{i+1}.

In the second round, each P_i broadcasts $v_i = k_{i,i+1} - z_{i-1,i}$. The values $v_1, v_2, \ldots, v_m$, constitute the following system of linear equations:

$$\begin{array}{rcrcrcrcl}
k_{1,2} & - & k_{2,3} & & & & & & = v_2 \\
 & & k_{2,3} & - & k_{3,4} & & & & = v_3 \\
 & & & & \vdots & & & & \\
 & & & & & & k_{m-1,m} & - & k_{m,1} = v_m \\
- & k_{1,2} & & & & & & & k_{m,1} = v_1
\end{array}$$

However, a system of linear equations on this form have infinitely many solutions in real number $\mathbb{R}$. Since $\mathbb{Z}_n^*$ is large, computing all solutions is therefore infeasible, and secret group key confidentiality is preserved.

To illustrate by a small example, the numbers v_i from the 3 users P_1, P_2, P_3, constitute the following system of linear equations:

$$\begin{array}{rcrcrcl}
 & k_{1,2} & - & k_{2,3} & & & = v_1 \\
 & & & k_{2,3} & - & k_{3,1} & = v_2 \\
- & k_{1,2} & & & & k_{3,1} & = v_3
\end{array}$$

By adding v_1 and v_3, we see that

$$\begin{array}{rcrcrcl}
k_{1,2} & - & k_{2,3} & & & = v_1 \\
 & & k_{2,3} & - & k_{3,1} & = v_2 \\
 & - & k_{2,3} & + & k_{3,1} & = v_1 + v_3
\end{array}$$

I.e., $v_1 + v_2 + v_3 = 1$. The system can therefore not be reduced, and has an infeasible number of solutions given that $\mathbb{Z}_n^*$ is large.□

Security Requirement 2 *The employed signature scheme must be secure.*

Proof. The signature scheme is analogous to the authentication scheme of [8] where the difficulty is to find two integers, x and y, so that $y_i^e \stackrel{?}{\equiv} ID_i \cdot x_i^{x_i} \pmod{n}$ holds. The signature in the second round is represented by

$$\begin{aligned} X_i &= \alpha^{f(ID_i)\cdot z_{i,i+1}} \pmod{n} \\ Y_i &= s_i \cdot h_i^{z_{i,i+1}\cdot f(X_i,v_i,c)} \pmod{n} \end{aligned}$$

where $z_{i,i+1}$ represents a value that may be known by the other participants. The value of d is unknown. Given a signature Y_i where the corresponding long-term secret user keys (s_i, h_i) are unknown to others than P_i, according to the Discrete Logarithm Problem, the factor $\alpha^{d\cdot f(ID_i)\cdot z_{i,i+1}\cdot f(X_i,v_i,c)}$ is computationally infeasible to obtain from Y_i since the value of d is unknown. Since s_i constitutes the other unknown factor of Y_i, it is thus infeasible to derive the secret user keys (s_i, h_i) from Y_i.

Since the factorization of n is unknown, it is correspondingly not feasible to obtain $ID_j^{(e^{-1})} \pmod{n}$.

Attempts to forge a valid signature Y_i by raising it to the power of a different context c'; $Y_i^{c'}$, will not provide a valid signature since (and as long) $ID_i^{c'}$ does not constitute a valid identity. □

A variant of this authentication scheme is found in [9] where the verification is according to

$$y_j^e \stackrel{?}{\equiv} ID_j \cdot x_j^{f(T)} \pmod{n}$$

where T is a time-stamp. This scheme is not resistant to the Extended Euclidian Algorithm attack [11]. If e and a are relatively prime, we can find two integers u, v, so that $e \cdot u = 1 + f(T) \cdot v$. An adversary can thus pick a valid ID_j, and compute $x_j = ID_j^v \pmod{n}$ and $y_j = ID_j^u \pmod{n}$. The attack succeeds in the scheme in [9] because

$$y_j^e = ID_j \cdot x_j^{f(T)} = (ID_j^u)^e = ID_j \cdot (ID_j^v)^{f(T)} \pmod{n}$$

In our scheme, this attack is thwarted, since $y_j^e \stackrel{?}{\equiv} ID_j \cdot x_j^{f(x_j,T_j)} \pmod{n}$ and x_j is used as self-verifying parameter by raising it to a power determined by itself as parameter to the hash-function.

Security Requirement 3 *The protocol must provide user key confidentiality.*

Proof. The number $Y_i = s_i \cdot h_i^{z_{i,i+1}\cdot f(X_i,v_i,c)} \pmod{n}$ contains two unknown factors where s_i is a secret user key. Since n is a large number, it is computationally infeasible to factorize Y_i and obtain s_i. We have that $f(X_i, v_i, c)$ is publicly available. In case s_i and $z_{i,i+1}$ were revealed, the secret user key h_i is still protected given Y_i due to the unknown factorization of n.

Also note that given a secret user key $h_i = \alpha^{d \cdot f(ID_i)} \pmod{n}$, it is computationally infeasible to derive α^d according to the Factorization Problem. This could be used to derive the new user key h_i. □

Security Requirement 4 *The protocol must provide forward secrecy.*

Proof. The values of the group keys depend on the Diffie-Hellman values $z_{i,i+1}$, $1 \leq i \leq m$, established by each of the participants for each session. The secret group key is the sum of $f(ID_i) \cdot z_{i,i+1}$, $1 \leq i \leq m$, where we consequently assume that any $z_{i,i+1}$ is unknown to others than the group. The secret group key is thus not directly linked to the long-term secret user keys (s_i, h_i).

Due to the Discrete Logarithm Problem, it is computationally infeasible to obtain the secret $z_{i,i+1}$ given $X_i = \alpha^{f(ID_i) \cdot z_{i,i+1}} \pmod{n}$. It is also computationally infeasible to obtain $z_{i,i+1}$ given $Y_i = s_i \cdot h_i^{z_{i,i+1} \cdot f(X_i, v_i, c)} \pmod{n}$ and (s_i, h_i) due to the Discrete Logarithm Problem. Forward secrecy is therefore achieved. □

Security Requirement 5 *The protocol must be secure against substitution attacks.*

Proof. Assuming that the employed signature scheme is secure, it is computationally infeasible for an adversary A' to successfully forge any signature Y_i' for other sessions. Suppose A' knows the values $k'_{i-1,i}$, $k'_{i,i+1}$, v_i', X_i' and Y_i' of a former session c'. Because $k'_{i-1,i}$ and $k'_{i,i+1}$ refer to a former session, A' cannot set $k_{i-1,i}$ and $k_{i,i+1}$ of a new session to equal the former $k'_{i-1,i}$ and $k'_{i,i+1}$ due to the contributory nature of DH key agreement.

Thus, A' can neither control that v_i' will have a valid value, nor the session value c' to be equal subsequent sessions c, since c is unique for each session, collaboratively determined by inputs of all participants. Since Y_i' is cryptographically locked to v_i' and c', replaying former messages will consequently be detected due to signature verification. Thus, the adversary cannot successfully replay and substitute messages with former ones. □

Security Requirement 6 *The proposed protocol must provide unique group keys for each session.*

Proof. Each user P_i contributes equally to the values of the group keys by the secret $k_{i,i+1}$. Assuming that at least one participant selects a new and unique random r_i, then $k_{i,i+1}$ and $k_{i-1,i}$ will be unique, thereby making the group key pair unique. Moreover, assuming that secure signature scheme withstands substitution attacks, no single user can enforce old group keys to be re-established. Thus, key freshness is achieved. □

5 Conclusion

We have proposed an efficient and practical public group key cryptosystem where groups containing any number of users can flexibly establish public/private key pairs that represent the groups. No online trusted third party is required neither for group key establishment nor group key certification. By means of public group keys, outsiders can address the group confidentially, and secure communication is provided internally among the group members. The signature scheme provides identity-based certification of public group keys and key establishment messages towards the corresponding group participants.

References

[1] Y. Desmedt. Society and group oriented cryptography: A new concept. Advances in Cryptology, Proc. of Crypto'87, LNCS, pp. 120 – 127, Springer-Verlag, 1988.

[2] Y. Desmedt, Y.Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307 – 315, Springer-Verlag, 1990.

[3] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, vol. 547, pp. 522 – 526, Springer-Verlag, 1991.

[4] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner, LNCS, vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[5] H. Ghodosi, S. Saeednia. A Modification to the Self-Certified Group-Oriented Cryptosystem Without a Combiner. Electronics Letters, vol. 37, no. 2, 2001.

[6] M. Burmester, Y. Desmedt. A secure and efficient conference key distribution system. In proc. of Eurocrypt'94, LNCS, vol. 950, pp. 275 – 286, Springer-Verlag, 1994.

[7] M. Just, S. Vaudenay. Authenticated multi-party key agreement. Proc. of ASIACRYPT'96, LNCS, vol. 1163, pp. 36 – 49, Springer-Verlag, 1996.

[8] K. Koyama, K. Ohta. Security of improved identity-based conference key distribution systems. Adv. in Cryptology - EuroCrypt '88, LNCS 330, pp. 11 – 19, Springer-Verlag, 1988.

[9] W. Yang, S. Shieh. Password authentication schemes with smart cards. Computer & Security, vol. 18, no. 8, pp. 727 – 733, 1999.

[10] E. Okamoto, K. Tanaka. Key Distribution System Based on Identification Information. IEEE Journal on Selected Areas in Communications, vol. 7, no. 4, pp. 481-485, 1989.

[11] K. Chen, S. Zhong. Attacks on the (enhanced) Yang-Shieh authentication. Computer & Security, vol 22, no. 8, pp. 725 – 727, 2003.

[12] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment. ISBN 3-540-43107-1, Springer-Verlag, 2003.

[13] M. Girault. Self-certified public keys. Advances in Cryptology - EUROCRYPT '91, LNCS 547, pp. 490 – 497, Springer-Verlag, 1991.

Paper C

Efficient Hierarchical Group-Oriented Key Establishment and Decryption

Sigurd Eskeland and Vladimir Oleshchuk

Efficient Hierarchical Group-Oriented Key Establishment and Decryption

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract
In this paper, we present three related and efficient cryptographic schemes for secure communication for hierarchically composed groups. The first provides secure establishment of hierarchically ordered session keys in agreement for hierarchically composed user groups. An essential security property is that users can only obtain hierarchical session keys for their own and underlying levels, while this is prevented for overlying levels. Moreover, in contrast to many existing hierarchical schemes, our scheme prevents that users of any class may deduce the long-term hierarchical keys of other classes.
This scheme is extended to a hierarchical public key cryptosystem based on the ElGamal cryptosystem, and furthermore to an ElGamal-based threshold decryption scheme. Due to the threshold security requirement, at least t arbitrary group members are required to carry out decryption. The threshold scheme requires only one round of broadcasting in the decryption phase, and is thus well-suitable for wireless networks.

1 Introduction

Secure group-oriented communication refers to communication that is facilitated in such a way that only the communicated data can only be intelligibly decoded by the participants of the pertaining group. Secure group-oriented communication is commonly achieved by means of group-oriented cryptographic protocols. There are several concepts and purposes relating to secure group communication [3]:

- Member identification and authentication. Authentication is important in order to prevent an intruder from impersonating a legitimate group member. Thus, authentication mechanisms must be used to allow an entity to verify whether another entity is really who it claims to be.
- Access control. After a party has been identified, access control is performed to validate group members before giving them access to some restricted or privileged information, e.g., some file or group communication.

- Generation and distribution of key material. It is necessary to change the key at regular intervals to safeguard its secrecy. Each key must be completely unrelated to previous and future keys, otherwise compromised keys may reveal other keys.

Secure key management plays an important role for providing secure communication among a number of parties, where the basic function is establishment and maintenance of secret key relationships between concerning and genuine parties.

A great variety of security protocols and methods for secure group communication has been proposed (e.g., see [1–3]), and common for most of them is that they are non-hierarchial. This is of course suitable for non-hierarchical groups where the group members have the same ranking. However, most organizations would in practice be hierarchically organized. Accordingly, team members would have various rankings according to their job positions in the organization, where individuals of lower ranking usually would be entrusted less confidential information than those of higher rankings. For example, in the medical scenario, medical care is given by medical teams that are composed of doctors and nurses. Due to the sensitive nature of personal medical data, it could be required that medical practitioners of lower rankings should be given access to less sensitive or confidential medical information than those of higher rankings. It is also essential that the medical data is transferred securely to the legitimate team members. This could be achieved by means of encryption which would require secure establishment of secret shared session keys in agreement with the user hierarchy.

In this paper, we present in Section 3 a cryptographic scheme for secure key establishment for hierarchical groups. The scheme is public key-oriented, allowing external parties to carry out encryptions using the hierarchical public keys. We assume that each participant is associated with one user class (or security class) that corresponds to his or her job position, and (for simplicity) that there is one security class for each hierarchical level. The scheme enables the participants to securely establish one secret hierarchical session key, subsequently referred to as *class key*, for each security class.

An essential security property is that the participants of any given security class can compute the secret class keys that are established by their own and underlying security classes, while it is computationally infeasible to obtain class keys of overlying security classes. In contrast to existing hierarchical access control schemes providing computation of long-term predefined hierarchical keys of underlying classes, our scheme is session-oriented and enables computation of hierarchical session keys, while disclosure of hierarchical long-term keys is prevented.

Each class is associated with a long-term public key, which the hierarchical class keys are computed as a function of. Using the public keys enable arbitrary individuals outside and inside the group to securely address any security class. Participants of the pertaining security classes are permitted to decrypt data pertaining to their own and underlying security classes, while it is prevented that ciphertexts pertaining to overlying security classes can be disclosed.

Section 4 presents a modification, incorporating the ElGamal public key cryptosystem into the key establishment scheme. In Section 5, the hierarchical key establishment scheme is extended to a hierarchical broadcast-oriented threshold decryption scheme. The participants broadcast only one message in the decryption phase, and it is therefore suitable for wireless networks.

2 Related work

In this section, we will attempt to give a brief overview of the field of hierarchical encryption and to point out how the security schemes presented in this paper conceptually differ from other hierarchical cryptographic schemes.

Hierarchical conference key agreement seems to be little investigated in the literature. Hwang et al. [15] proposed a hierarchical key agreement protocol that enables establishment of keys for each user class. Moreover, the participants of a given class can obtain the hierarchical session keys (i.e., class keys) of the underlying security classes, while it is prevented that class keys of overlying security classes can be obtained. A major disadvantage is that it is highly inefficient because the number of rounds equals the number of participants. A more efficient hierarchical key agreement protocol that requires only two rounds of broadcasting was proposed in [17].

Hierarchical access control is a class of cryptographic schemes that supports establishment and deduction of long-term predefined cryptographic keys that comply with some hierarchical user structure (e.g., see [18–21]). This allows users of a given security class to compute securely such keys associated with their own and underlying security classes, while computation of keys associated with overlying security classes is prevented. While the hierarchical key establishment schemes presented in this paper provide secure ad-hoc establishment of a number of hierarchically arranged sessions keys, HAC schemes do in contrast enable a group of hierarchically ranked users to deduce hierarchical predefined static keys. Computation of such hierarchical long-term predefined keys in contrast to hierarchical short-term sessions keys is a considerable limitation to the applicability and usefulness of such schemes.

Due to access control purposes, most HAC schemes are compliant with user dynamics, i.e., inclusion and exclusion of users and corresponding renewal of hierarchical keys for the pertaining security classes.

A class of hierarchical encryption schemes that is related to HAC is hierarchical identity-based encryption (HIBE). See [22–24] for examples of such schemes. As with identity-based encryption (IBE), a participant is publicly represented by an identity, and has a private long-term key that is computed by a trusted party. HIBE schemes are hierarchical concerning the establishment of the long-term private keys. A motivation of HIBE seems to be to provide a distributive way for computation of private user keys in large organizations. A root authority computes the necessary public parameters for each security class. This allows users or authorities at a given class to compute the private keys of the underlying

classes. Like HAC, HIBE correspondingly entails the downside that private user keys are not prevented from being deduced, since it could be desirable that such keys could be used for other purposes as well.

Tree-based key management schemes (e.g., see [25–27]) can be regarded as centralized key distribution where the users of a group establish a *key tree*. The users constitute the leaf nodes of the tree, and the tree structure allows them to obtain the commonly established key that is located at the root. Thus, such schemes are non-hierarchical due to that all participants obtain the same shared secret key.

Scheme 1 differs from the above because it is not a key agreement scheme while it pertains secure establishment of session keys. We have not been able to find any hierarchical threshold decryption schemes in the literature.

3 Hierarchical centralized key distribution

It is common that corporations and organizations are hierarchically structured, where the ranking of the personnel would be defined according job positions. Also relevant is that many tasks are carried out by teamwork. For example, in the medical scenario, medical care is provided by medical teams that are composed of doctors and nurses, where the doctors have a higher ranking than the nurses. Due to the sensitive nature of personal medical data, it could be required that medical practitioners of lower rankings should only be privileged access to less sensitive or confidential medical information than those of higher rankings. This assumes that the medical data in electronic patient records is appropriately categorized according to sensitively levels, so that the sensitivity levels are in agreement with the hierarchical ranking of the medical personnel.

In this section, we present a hierarchical key distribution scheme for secure distribution of confidential data to hierarchical teams. For simplicity, we assume that there is only one class (or security class) of users for each hierarchical level or security level. The proposed scheme can easily be extended to a partially ordered hierarchical structure, i.e., hierarchical structures of several classes for each level.

3.1 Security requirements

The security requirements of the key establishment scheme are as follows:

Security Requirement 1. *Class key confidentiality.* Only legitimate participants must be able to compute the hierarchical session keys (or class keys). This property is associated with implicit authentication of the users according to their security class.

Security Requirement 2. *Implicit authentication.* It must be ensured that only the proper users get the pertaining hierarchical session keys and in agreement with their security classes.

Security Requirement 3. *User key confidentiality.* It must be prevented that long-term hierarchical user keys can be disclosed to outsiders and to members of others security classes. Although the participants of a given class can compute the hierarchical session keys of underlying security classes, it must be prevented that participants of any given class can deduce secret long-term hierarchical user keys. This is essential since such keys could be used for other applications whereof participants of other security classes may not be involved with.

Security Requirement 4. *Onewayness.* Participants of any given security class can only compute the hierarchical session keys (class keys) pertaining to their own and underlying security classes, while it is prevented that class keys pertaining to overlying security classes can be deduced.

3.2 Scheme 1

Assumptions. Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ be a team of n participants where each participant is associated with one hierarchical level. Let $\mathcal{U}$ be divided into λ disjoint security classes $S_i \subseteq \mathcal{U}$, i.e., $S_i \cap S_j = \emptyset$, $i \neq j$, where $i, j \in \{1, \ldots, \lambda\}$, so that each level contains one security class. The security classes are totally ordered so that $S_i \prec S_j$ if $i < j$. Let $S_\lambda \subseteq \mathcal{U}$ constitute the *top* security class.

Initializations. A trusted authority (TA) is required to set up the scheme by providing a secret hierarchical key and a public key for each respective security class. Initially, the TA selects a large public prime $p = 2 \cdot q + 1$ where q is also prime. The TA selects a generator (or primitive element) α to $\mathbb{Z}_q$ [4, p. 30].

The TA randomly generates a long-term secret key $k_j \in \mathbb{Z}_q$ for each security class $S_j \subseteq \mathcal{U}$. These are securely transferred to the users in the respective classes. The TA computes the public parameters $y_j = \alpha^{k_j} \pmod p$, $1 \leq j \leq \lambda$.

Key establishment. One particular party is required to initiate the protocol. We refer to this as the registry. Note that the registry is a semi-trusted party in the sense that it is capable of computing the secret class keys due to the knowledge of the secret number r in Step 1.

Step 1. The registry generates a random secret number r from $\mathbb{Z}_q$, and computes and broadcasts

$$Z = \{z_j = y_j^r - \alpha^{(y_{j+1}^r)} \pmod p \mid 1 \leq j < \lambda\}$$

and

$$R = \alpha^r \pmod p$$

Step 2. Each participant in $S_i \subseteq \mathcal{U}$ computes the class key for his or her own security class S_i according to

$$K_i = R^{k_i} \pmod p$$

and for the underlying security classes S_j, $1 \leq j < i \leq \lambda$, according to the recurrence relation

$$K_j = z_j + \alpha^{K_{j+1}} \pmod p$$

Example. To illustrate, in this example there are 3 security classes S_1, S_2, S_3. The registry broadcasts $z_1 = y_1^r - \alpha^{(y_2^r)}$, $z_2 = y_2^r - \alpha^{(y_3^r)}$, $R = \alpha^r$. The participants of S_3 computes recursively $K_3 = R^{k_3}$, $K_2 = z_2 + \alpha^{K_3}$, $K_1 = z_1 + \alpha^{K_2}$, where all computations are in $\mathbb{Z}_p$.

3.3 Security analysis

In this section, we provide a security analysis showing that the presented scheme is secure in agreement with the security requirements presented in Section 3.1. (We assume that all computations are in $\mathbb{Z}_p$.)

Security Requirement 1. Class key confidentiality. The confidentiality of class keys is based on the Diffie-Hellman Computational Problem, meaning that given α^x and α^y, it is computationally infeasible to find $\alpha^{x \cdot y}$. Public values are $R = \alpha^r$ and $y_i = \alpha^{k_j}$, $1 \leq j \leq \lambda$. Accordingly, it is computationally infeasible to compute $K_j = \alpha^{r \cdot k_j}$.

The register publicizes $z_j = K_j - \alpha^{K_{j+1}}$, where Y_j and K_{j+1} (and hence $\alpha^{(K_{j+1})}$) are unknown.

Let $z'_j = K_j - K_{j+1}$ where $1 \leq i < \lambda$. Then λ values z_j form a linear equation system. For example, if $\lambda = 3$, we have that

$$\begin{array}{rlll} K_1 - & K_2 & & = z'_1 \\ & K_2 - & K_3 & = z'_2 \\ & & K_3 - K_4 & = z'_3 \end{array}$$

Since λ such equations contain $\lambda + 1$ unknowns and , the equation system cannot be solved, since the number of solutions in $\mathbb{Z}_p$ would be computationally infeasible, since p is a large prime. Since K_j cannot be deduced from a system based on the form $z'_j = K_j - K_{j+1}$, likewise K_j cannot be deduced from a system on the form $z_j = K_j - \alpha^{K_{j+1}}$.

Security Requirement 2. Implicit authentication. Thus, K_j can only be correctly established by means of the proper k_j, and then subsequently for K_i, $1 \leq i \leq j$. This provides accordingly implicit authentication of the users according to their security class.

Security Requirement 3. User key confidentiality. The public keys are computed according to $y_j = \alpha^{k_j}$. No other numbers computed by means of the hierarchical user keys are broadcasted. Due to the Discrete Logarithm Problem, the hierarchical user keys k_j are protected from disclosure.

Security Requirement 4. Onewayness. The one-way security property prevents that participants in S_i can obtain K_j if $i < j$. Given $z_j = K_j - \alpha^{K_{j+1}}$, where K_j is known to the participants in S_j, it is easy to compute $\alpha^{K_{j+1}}$. However, it is computationally infeasible to compute K_{j+1} in $\mathbb{Z}_p$ given $\alpha^{K_{j+1}}$ due to the Discrete Logarithm Problem.

4 A hierarchical ElGamal cryptosystem

In this section, we apply the key establishment scheme from the previous section for a hierarchical variant of the ElGamal public key cryptosystem [11]. An essential security property of this scheme is onewayness, so that participants of any given security class can only carry out decryption of data that is targeted to their own and underlying security classes, while it is prevented that they can decrypt data targeted to overlying security classes.

4.1 Scheme 2

Assumptions and Initializations. See Section 3.2. In addition to α, the TA selects one more generator (or primitive element) β to $\mathbb{Z}_q$.

Encryption. By means of the public keys, any individual (the sender) can encrypt data for each security class of $\mathcal{U}$. The sender randomly generates the secret number r from $\mathbb{Z}_q$, and encrypts the message m_j, if any, for the security classes $S_i \subseteq \mathcal{U}$, $1 \leq i \leq j$, according to

$$C = \{c_j = m_j \cdot \beta^{-Y_j} \pmod{p} \mid 1 \leq j \leq \lambda\}$$

and

$$Z = \{z_j = Y_j - \alpha^{Y_{j+1}} \pmod{p} \mid 1 \leq j < \lambda\}$$

and

$$R = \alpha^r \pmod{p}$$

where $Y_j = y_j^r$. Then (C, Z, R) is sent to $\mathcal{U}$. Note that β^{Y_j} corresponds to the secret encryption factor of the ElGamal scheme.

Decryption. In order for the participants of S_i to decrypt the cryptograms pertaining to its own and underlying classes S_j, $1 \leq j \leq i \leq \lambda$, the secret encryption factor β^{Y_j} has to be recovered for each of the pertaining classes in agreement with the computations of class keys in the previous section.

Each participant in $S_i \subseteq \mathcal{U}$ computes

$$Y_i = R^{k_i} \pmod{p}$$

and for the underlying security classes S_j, $i < j$, according to the recurrence relation

$$Y_j = z_j + \alpha^{Y_{j+1}} \pmod{p}$$

Finally, the plaintexts for the relevant classes are restored according to

$$m_j = c_j \cdot \beta^{Y_j} \pmod{p}$$

in agreement with the ElGamal public key cryptosystem.

4.2 Security remarks

In contrast to computing the cryptotext according to $c_j = m_j \cdot Y_j$, in full agreement with the ElGamal scheme, it is in this scheme computed as $c_j = m_j \cdot \beta^{-Y_j}$. The reason for introducing the additional generator β is to obstruct "guessing" attacks.

If the cryptotext is computed according to $c_j = m_j \cdot Y_j$, we have the following security problem: If the corresponding plaintext m_j of a given cryptotext c_j is known by some outsider or by a participant of an underlying security class, this user could compute the encryption factor as $Y_j = \frac{c_j}{m_j}$. The user could then easily compute the encryption factors for the underlying levels.

This attack is obstructed in the presented scheme, since this would result in $\beta^{Y_j} = \frac{c_j}{m_j}$. Due to the Discrete Logarithm Problem, it would be computationally infeasible to obtain Y_j given β^{Y_j} in Z_p, since p is a large prime.

The essential security properties of this cryptosystem are onewayness, user key confidentiality, implicit authentication and data confidentiality. The data confidentiality security property is equivalent to the secrecy of the encryption factors β^{Y_i}. Since establishment of the encryption factors corresponds to establishment of class keys in the previous scheme, the data confidentiality is consistent with the security of the ElGamal cryptosystem, and with the security properties of class key confidentiality and implicit authentication in Section 3.3. Also see Section 3.3 for security analysis of the onewayness security property and user key confidentiality.

5 Broadcast-oriented hierarchical threshold decryption

In a number of scenarios, it could be desirable or required that certain actions should be carried out on the condition of a group consensus. Threshold-oriented cryptosystems enforce such requirements in agreement with the consensus of a minimum, arbitrary number of participants from a group or organization in order to carry out a relevant cryptographic computation. The minimum required number of participants is known as the threshold number t. In contrast, requiring the consensus and collaboration of a predefined, fixed set of participants to carry out such a cryptographic computation would be inconvenient, inflexible, and mostly inapplicable if one or more of the designated users are absent. The concept of threshold cryptography may also comply to scenarios where some sort of separation of duty is required, and where individuals should be prevented to carry out such computations on their own.

An example of such application could be where the holder or originator of some sensitive information like a secret key, is only willing to disclose it as result of the consensus of a given number of designated participants. A practical example could be access to a bank vault, where it would not be desirable that one person

alone would possess and control the key to the vault due to the risk of fraud, robbery and extortion. A preferable solution could be a threshold-oriented lock, requiring for instance at least 3 employees out of 5, each holding a unique and secret key, in order to unlock the vault.

Threshold cryptosystems are commonly based on the Shamir secret sharing scheme [10], and on the concept of public key cryptography. In threshold decryption, anyone can confidentially encrypt messages to a group that is publicly represented by their public key. Such cryptograms can only be decrypted due to the collaboration of at least t participants of the group. Examples of such can be found in [5–7]. Likewise, threshold signatures schemes convey that only a minimum subset of the team can compute signatures due to the threshold requirement, e.g., see [8, 9].

In the following subsection, we present a hierarchical threshold-decryption scheme based on the hierarchical scheme in the previous subsection. In the threshold-variant of this scheme, decryption can only be carried out due to the collaboration of at least t participants of the team $\mathcal{U}$. It is broadcast-oriented in the decryption phase, and since each user broadcasts only one message, it is well-suitable for wireless networks.

5.1 Scheme 3

Assumptions. See Section 3.2.

Initialization. A trusted authority (TA) is required to set up the scheme by providing a secret hierarchical key and a public key for each respective security class. Initially, the TA selects a large public prime $p = 2 \cdot q + 1$ where q is also prime. The TA selects two generators α and β to $\mathbb{Z}_q$.

The basis for the threshold mechanism is Shamir secret sharing scheme [10] where a secret polynomial of order t defines the threshold requirement. The TA defines the threshold requirement (t, n), where $\lambda \leq t \leq n$. The TA randomly generates the polynomial coefficients a_j of the polynomial

$$f(x) = \sum_{j=0}^{t-1} a_j \, x^j$$

The TA randomly generates λ long-term keys $k_i \in \mathbb{Z}_q$ for each security class $S_j \subseteq \mathcal{U}$ which the TA securely hands each user in the respective classes. The TA computes the public keys

$$y_j = \alpha^{x_j} \pmod{p}$$

where $x_j = a_{j-1} \cdot k_j$, $1 \leq j \leq \lambda$.

The TA computes for each $P_i \in \mathcal{U}$ a secret user share

$$s_i = f(i) \pmod{q}$$

Encryption. In agreement with Section 4, any individual (a sender) can encrypt data for any security class by means of the public keys. The sender generates a random secret number $r \in \mathbb{Z}_q$, and encrypts a message m_j, if any, for each security class $S_j \subseteq \mathcal{U}$ according to

$$C = \{c_j = m_j \cdot \beta^{(-y_j^r)} \pmod p \mid 1 \leq j \leq \lambda\}$$

and

$$Z = \{z_j = y_j^r - \alpha^{(y_{j+1}^r)} \pmod p \mid 1 \leq j < \lambda\}$$

and

$$R = \alpha^r \pmod p$$

and then sends (C, Z, R) to $\mathcal{U}$. Note that each $c_j \in C$ can be decrypted by the participants in the given and underlying security classes $S_i \subseteq \mathcal{U}$, $1 \leq i \leq j$.

Decryption. Due to the threshold requirement, the computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t team members are required in order to restore the plaintexts. Each participant $P_i \in T$ computes and broadcasts

$$w_i = R^{s_i} \pmod p$$

Let $I_T = \{i \mid P_i \in T\}$. Lagrange interpolation is given by

$$f(x) = \sum_{i \in I_T} s_i \, b_i(x) \quad \text{where} \quad b_i(x) = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x - j}{i - j} = \sum_{j=0}^{t-1} c_{i,j} \, x^j$$

By means of the Lagrange basis coefficients $c_{i,j}$, we can compute the polynomial coefficients a_j of $f(x)$:

$$f(x) \;=\; \sum_{i \in I_T} s_i \, b_i(x) = \sum_{i \in I_T} s_i \, (\sum_{j=0}^{t-1} c_{i,j} \, x^j) = \sum_{j=0}^{t-1} (\sum_{i \in I_T} s_i \, c_{i,j}) x^j = \sum_{j=0}^{t-1} a_j \, x^j$$

Thus,

$$a_j = \sum_{i \in I_T} s_i \, c_{i,j} \qquad (0 \leq j \leq t-1)$$

By means of Lagrange interpolation, the following λ numbers must be computed:

$$A_j = \prod_{i \in I_T} w_i^{c_{i,j}} = \prod_{i \in I_T} R^{s_i \cdot c_{i,j}} = R^{a_j} \pmod p \qquad (0 \leq j \leq \lambda - 1)$$

(Since these numbers are publicly available, the computation could be performed by a single party that subsequently broadcasts the results.)

Each participant in $S_i \subseteq \mathcal{U}$ computes

$$Y_i = A_{i-1}^{k_i} = R^{a_{i-1} \cdot k_i} = R^{x_i} \pmod p$$

The participant then computes Y_j, $i > j$, for the underlying security classes (if any) in agreement with the recurrence relation

$$Y_j = z_j + \alpha^{Y_{j+1}} \pmod{p}$$

Finally, the plaintexts for the relevant classes are restored according to

$$m_j = c_j \cdot \beta^{Y_j} \pmod{p}$$

Example. Let us consider a group $\mathcal{U}$ consisting of two (disjoint) security classes S_1 and S_2 to illustrate. All computations are in $\mathbb{Z}_p$.

Encryption. The sender encrypts and broadcasts $c_1 = m_1 \cdot y_1^r$, $c_2 = m_2 \cdot y_2^r$, $z_1 = y_1^r - \alpha^{(y_2^r)}$, $R = \alpha^r$.

Decryption. At least t participants from $\mathcal{U}$ compute and broadcast w_i. The participants of S_2 compute by Lagrange interpolation $A_1 = R^{a_1}$, and then $Y_2 = A_1^{k_2}$, $Y_1 = z_1 + \alpha^{Y_2}$. Finally, the plaintexts are restored by $m_1 = c_1 \cdot Y_1$, $m_2 = c_2 \cdot Y_2$.

5.2 Security remarks

The essential security properties of the threshold cryptosystem is in agreement with the schemes presented in Section 3 and Section 4. In addition to this, we have the threshold requirement which is provided by the Shamir secret sharing scheme that is incorporated into this scheme.

6 Conclusion

In this paper, we have presented three related efficient broadcast-oriented cryptographic schemes for secure communication for hierarchically composed groups. The first is a hierarchical key establishment scheme. The second scheme is a hierarchical public key cryptosystem based on the presented hierarchical key establishment scheme and the ElGamal cryptosystem. This is further extended to a threshold decryption scheme, which imposes a threshold security requirement, requiring the participation of at least t group members in order to carry out decryption.

References

[1] X. Zou, B. Ramamurthy, S. Magliveras. Secure group communications over data networks. ISBN 0-387-22970-1. Springer Science-Business Media, Inc., 2005.

[2] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment. ISBN 3-540-43107-1, Springer-Verlag, 2003.

[3] S. Rafaeli, D. Hutchinson. A survey of key management for secure group communication. ACM Computing Surveys, Vol.35, No.3, pp. 309–329, 2003.

[4] J. Pieprzyk, T. Hardjono, J. Seberry. Fundamentals of computer security. ISBN 3-540-43101-1, Springer-Verlag, 2003.

[5] Y. Desmedt, Y. Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307–315, Springer-Verlag, 1990.

[6] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, Vol. 547, pp. 522–526, Springer-Verlag, 1991.

[7] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner. LNCS, Vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[8] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings. Vol. 141, No. 5, pp. 307-313, 1994.

[9] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders, Eurocrypt 1994, pp. 194–204.

[10] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612–613, 1979.

[11] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469–472, 1985.

[12] C. Li, J. Pieprzyk. Conference Key Agreement from Secret Sharing. ACISP '99, Springer-Verlag, pp. 64–76, 1999.

[13] J. Pieprzyk, C. H. Li. Multiparty key agreement protocols. IEE Proceedings, Computer and Digital Techniques, Vol. 147, No. 4, pp. 229–236, 2000.

[14] R. Rivest, A. Shamir, L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Comm. of the ACM, Vol. 21, No. 2, pp. 120 – 126, 1978.

[15] M. Hwang, W. Tzeng. A conference key distribution scheme in a totally-ordered hierarchy. ICOIN 2003, LNCS 2662, pp. 757 – 761, Springer-Verlag, 2003.

[16] S. Eskeland. Efficient Hierarchical Conference Key Establishment in Wireless Networks. IASTED International Conference on Communication, Network and Information Security '05, pp. 94 – 98, Acta Press, 2005.

[17] S. Eskeland, V. Oleshchuk. Hierarchical Multi-Party Key Agreement for Wireless Networks. Third International Symposium on Information Assurance and Security '07, pp. 39 – 43, IEEE Computer Society, 2007.

[18] F. Kuo, V. Shen, T. Chen, F. Lai. Cryptographic key assignment scheme for dynamic access control in a user hierarchy. IEE Proc. Computers & Digital Techniques, Vol 146, No. 5, 1999, pp. 235 – 240.

[19] C. Lin. Dynamic key management schemes for access control in a hierarchy. Computer communications, Vol. 20, No. 15, pp. 1381 – 1385, 1997.

[20] C. Chang, C. Lin, W. Lee, P. Hwang. Secret sharing with access structures in a hierarchy. AINA, Vol. 2, pp. 31 – 34, 2004.

[21] X. Zou, B. Ramamurthy, S. Magliveras. Chinese Remainder Theorem based hierarchical access control for secure group communications. ICICS, LNCS, Vol. 2229, pp. 381 – 385, 2001.

[22] C. Gentry, A. Silverberg. Hierarchical ID-Based cryptography. In Proceedings of Asiacrypt'02, pp. 548 – 566, 2006.

[23] U. Hengartner, P. Steenkiste. Exploiting Hierarchical Identity-Based Encryption for Access Control to Pervasive Computing Information. SECURECOMM '05: Proc. of the First Int. Conf. on Security and Privacy for Emerging Areas in Comm. Networks, pp. 384–396, IEEE Computer Society, 2005.

[24] D. Boneh, X. Boyen, E. Goh. Hierarchical identity based encryption with constant size ciphertext. In Proceedings of Eurocrypt '05, LNCS, Vol 3493, pp. 440–456, Springer, 2005.

[25] Y. Kim, A. Perrig, G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. Proc. of 7th ACM CCS, pp. 235 – 244, 2000.

[26] L. Dondeti, S. Mukherjee, A. Samal. DISEC: a distributed framework for scalable secure many-to-many communication. Proc. of 5th IEEE ISCC, pp. 693 – 698, 2000.

[27] A. Sherman, D. McGrew. Key establishment in large dynamic groups using one-way fucntion tree. IEEE transactions on Software Engineering, Vol. 29, No. 5, pp. 444 – 458, 2003.

Paper D

Collusion-Resistant Threshold Decryption

Sigurd Eskeland and Vladimir Oleshchuk

Collusion-Resistant Threshold Decryption

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract

Most (t, n) threshold-oriented cryptosystems incorporate the polynomial-based (t, n) threshold secret sharing scheme of Shamir. This makes them vulnerable to a collusion problem in the following ways: 1) A set of t colluding participants can compute the shared secret (i.e., a secret polynomial coefficient). Any participant holding the shared secret can subsequently carry out group-oriented threshold-oriented computations *individually*, thereby bypassing the threshold security requirement. 2) A set of t participants can moreover deduce all the secret polynomial coefficients which enables establishment of new user shares. In this paper, we propose a method applied to the threshold decryption scheme of Desmedt and Frankel that prohibits colluding participants to deduce any of the secret coefficients of the underlying threshold Shamir secret sharing scheme.

1 Introduction

Threshold cryptosystems is a class of group-oriented cryptosystems that is appropriate for enforcing group consensus. According to the threshold security requirement, the partial computations of t arbitrarily users of a coalition of n (where $t \leq n$) are required to carry out a given threshold computation. The threshold property provides flexibility since it allows an arbitrary composed subgroup of t participants, in contrast to requiring all, i.e., a fixed set of participants, for such computations. Accordingly, it prevents that single individuals can carry out such computations on their own.

A practical example of this could be access to a bank vault. It would not be desirable that one person alone would control the key to the vault due to the risk of fraud, robbery and extortion. Rather, the vault should only be unlockable due to consensus of at least t bank employees. However, since absent employees would be prohibited from participating, a practical and flexible vault locking mechanism could be based on the threshold property, and open as a function of the secret keys of t arbitrary employees of certain clearance.

Typical threshold-oriented applications are threshold decryption and threshold signatures. Threshold decryption cryptosystems enforce that a minimum number

of t arbitrary participants of a group of n participants are required to carry out decryptions pertaining to their group. Represented by a public key, outsiders can confidentially address the group. Only by collaboration in such a way where the active group members are providing partial computations, the encrypted message can be decrypted [1–3]. Likewise, regarding threshold signatures [4, 5], only a minimum subcoalition of the team can compute digital signatures due to the threshold requirement. Note that threshold signatures do not reveal the identity of the actual signing group members.[1] Another application is conference key establishment [6, 7]. The term threshold cryptography is sometimes used interchangeably with group-oriented cryptography, but the latter term has a more general meaning in the sense that group-oriented cryptosystems are not necessarily confined to threshold-oriented cryptosystems.

Threshold cryptosystems are in general based on a threshold-oriented secret sharing mechanism. The Shamir threshold secret sharing scheme [8] is by far most applied in such cryptosystems. Each group member is confidentially handed a secret user share from the trusted center that sets up the scheme. However, a potential problem is that a minimum coalition of t participants may collude and compute the shared secret. This would enable each of the colluders to subsequently bypass the threshold mechanism and to *individually* carry out the threshold computations, in violation of the threshold security property. The coalition can moreover compute all the secret polynomial coefficients, enabling them to illegitimately establish new user shares. If the threshold is low, this could be an imminent problem.

In this paper, we propose a modification to the threshold decryption scheme of Desmedt and Frankel [1] that prevents the polynomial coefficients of the user shares from being deduced. To our knowledge, the collusion problem in the context of threshold cryptosystems has not been previously addressed in the literature.

1.1 Threshold secret sharing

In the (t, n) threshold secret sharing scheme of Shamir [8], a secret number is split into n secret user shares. The secret can only be reconstructed by means of an arbitrarily composed subcoalition of a minimum number of t user shares. A trusted authority (TA) sets up the scheme by generating a secret polynomial of order $(t-1)$:

$$f(x) = \sum_{j=0}^{t-1} a_j \cdot x^j$$

Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ denote a team of n participants. For a given team $\mathcal{U}$, the TA arbitrarily selects a set of user inputs

$$A = \{x_j \mid P_j \in \mathcal{U}\} \subseteq \mathbb{Z}_q$$

[1] Note the distinction from group signatures (that are not threshold-based) where a single member signs on behalf of a group in such a way that the identity of the signer cannot be determined, only that the signer is member of a specific group.

where q is a large prime, and computes for each $P_i \in \mathcal{U}$ a secret user share according to

$$s_i = f(x_i) \pmod{q}$$

where $x_i \in A$. Normally, the shared secret is defined as $f(0) = a_0$ but can in general be $f(x)$ for any chosen $x \in \mathbb{Z}_q$. Any coalition $T \subseteq \mathcal{U}$ of (at least) t participants can obtain $f(x)$ for any $x \in \mathbb{Z}_q$ by Lagrange interpolation. Also note that t long-term shares make up a linear equation system:

$$\begin{pmatrix} 1 & x_1 & \dots & x_1^{t-1} \\ 1 & x_2 & \dots & x_2^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_t & \dots & x_t^{t-1} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{t-1} \end{pmatrix} = \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{t-1} \end{pmatrix}$$

Accordingly, any collusion of t participants can obtain the secret polynomial coefficients by solving the equation system, and moreover compute new shares and reconstruct already issued secret shares. The smaller t is, for instance 2 or 3, the more imminent the collusion problem may be.

1.2 A basic threshold decryption scheme

In this subsection, we illustrate the collusion problem by the basic threshold decryption scheme proposed by Desmedt and Frankel [1], which in turn is based on the ElGamal public key cryptosystem [9] and Shamir secret sharing [8].

A trusted authority (TA) sets up the scheme by computing the secret user shares for a team $\mathcal{U}$ according to Shamir secret sharing scheme presented in the previous subsection. The TA selects another large public prime p where $p = 2 \cdot q + 1$ and q is also prime. The TA moreover selects a generator α to $\mathbb{Z}_q$. The team is represented by the public key

$$y = \alpha^{-a_0} \pmod{p}$$

where $a_0 = f(0)$ is the shared secret, i.e, a secret polynomial coefficient of $f(x)$.

Encryption. By means of the public key y, a sender can encrypt a message m according to the ElGamal public key cryptosystem. The sender generates a random secret number r from $\mathbb{Z}_q$ and computes the cryptogram (c, R) where

$$c = m \cdot y^r \pmod{p} \quad \text{and} \quad R = \alpha^r \pmod{p}$$

and sends (c, R) to $\mathcal{U}$.

Decryption. To decrypt (c, R), the partial computations of a subcoalition T of the team $\mathcal{U}$ of at least t participants are required. Each $P_i \in T$ computes and sends

$$Y_i = R^{s_i} \pmod{p}$$

confidentially, i.e., via *secure channels*, to the other participants (or alternatively via secure channels to a trusted party). After having received the partial computations, the plaintext is restored according to Lagrange interpolation:

$$m = c \cdot \prod_{j \in I_T} Y_j^{b_j} \pmod{p}$$

where

$$b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x_j}{x_j - x_i} \pmod{q}$$

and $I_T = \{i \,|\, P_i \in T\}$. This corresponds to $m = (m \cdot \alpha^{-a_0 r}) \cdot \alpha^{r a_0}$.

One of the main purposes of threshold-oriented cryptography is to enforce collaboration between some entities in order to do a cryptographic computation. However, a coalition T can apply Lagrange interpolation directly on their secret user shares to obtain the shared secret $a_0 = \sum_{j \in I_T} s_j \cdot b_j \pmod{q}$. The members of the coalition are now able to *individually* decrypt all preceding and future cryptograms pertaining to the team by applying a_0 directly to obtain m as

$$m = c \cdot R^{a_0} \pmod{p}$$

and thus, bypassing the threshold requirement.

1.3 Security requirements

In order to provide collusion-resistance, the following security requirements must be maintained:

Security Requirement 1. *Protection of the secret coefficients.* Disclosure of one or more polynomial coefficients from $f(x)$ given any number of user shares must be prevented.

Security Requirement 2. *Protection of the user shares.* Disclosure of existing or computation of new user shares given any number of user shares must be prevented.

Security Requirement 3. *Protection of encrypted data.* The cryptographic security must be in agreement with the underlying threshold cryptosystem.

The last security property pertains to the security of the actual underlying cryptosystem.

2 Collusion-resistant threshold decryption

In this section, we propose a modification of the classical threshold decryption scheme of Desmedt and Frankel [1] that prohibits deduction of the shared secret or any of the secret polynomial coefficients from $f(x)$ given any number of user

shares, thus preventing that the threshold requirement can be bypassed without the application of at least t user shares. Thus, it prevents violation of the threshold requirement and moreover computation of new user shares.

2.1 Initialization

Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ denote a team of n participants. According to the number of participants of $\mathcal{U}$, the trusted authority (TA) randomly generates a set

$$K = \{k_1, \ldots, k_m\} \subseteq \mathbb{Z}_q$$

of $m = \lceil \log_2 n \rceil$ secret numbers and where q is a large prime. The TA selects another large public prime p such that $p = 2 \cdot q + 1$ and a generator α to $\mathbb{Z}_q$, and computes the corresponding set of public parameters

$$B = \{t_j \mid t_j = \alpha^{k_j} \pmod{p} \quad \text{and} \quad k_j \in K\}$$

for public representation of $\mathcal{U}$.

Let $(b_m, b_{m-1}, \ldots, b_1)$ denote binary representation of the user index i. Let

$$I_i(j) = b_j$$

represent bit j of the user index i. Then, the TA randomly selects a secret polynomial

$$f(x) = \sum_{j=0}^{t-1} a_j \cdot x^j$$

and computes for each $P_i \in \mathcal{U}$ a secret user share according to

$$s_i = f(i) \cdot w_i^{-1} \pmod{q}$$

where $w_i = \sum_{j=1}^{m} k_j \cdot I_i(j)$ and $k_j \in K$.

The team is represented by the public key $y = \alpha^{-a_0} \pmod{p}$ where $a_0 = f(0)$ is the shared secret. Note that since the elements of K are unknown, any coalition $T \subseteq \mathcal{U}$ is prohibited to reconstruct the coefficient a_0.

2.2 Encryption

By means of the public key y, a sender can encrypt a message m according to the ElGamal public key cryptosystem. The sender generates a random secret number r from $\mathbb{Z}_q$ and computes the cryptogram (c, Z) where

$$c = m \cdot y^r \pmod{p}$$

and

$$Z = \{z_j \mid z_j = t_j^r \pmod{p} \quad \text{and} \quad t_j \in B\}$$

and sends (c, Z) to $\mathcal{U}$.

2.3 Decryption

To decrypt the cryptogram, the partial computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required. Each $P_i \in T$ computes and sends

$$Y_i = \big(\prod_{j \in I_D} z_j^{I_i(j)} \big)^{s_i \cdot b_i} \pmod{p}$$

where

$$b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x_j}{x_j - x_i} \pmod{q}$$

and $I_T = \{i \,|\, P_i \in T\}$ and $I_D = \{i \,|\, z_i \in Z\}$ through *secure channels* to the other participants. By confidentially sharing their partial computations, the participants compute the secret encryption factor

$$Y = \prod_{j \in I_T} Y_j \pmod{p}$$

that corresponds to $Y = \alpha^{a_0 \cdot r}$. Subsequently, the plaintext is restored according to

$$m = c \cdot Y \pmod{p}$$

in agreement to the ElGamal public key cryptosystem [9].

2.4 Security analysis

In this section, we provide a security analysis showing that the presented scheme is secure in agreement with the security requirements presented in Section 1.3.

Security Requirement 1. Protection of the secret coefficients. Each user share is computed on the form $s_i = f(i) \cdot w_i^{-1}$ where the factors $f(i)$ and w_i are unique and unknown to each user $P_i \in \mathcal{U}$. If w_i is the same for all users, any coalition of t users could obtain the secret coefficients by using their secret shares to solve the linear equation system formed by their user shares. However, since the values of w_i, $1 \leq i \leq n$, are distinct, the coefficients of the secret user shares $(s_1, \ldots, s_n)$ form a set of Diofant equations

$$\begin{aligned} b_{1,0} + x_1\, b_{1,1} + \ldots + x_1^{t-1}\, b_{1,t-1} &= s_1 \\ &\vdots \\ b_{n,0} + x_n\, b_{n,1} + \ldots + x_n^{t-1}\, b_{n,t-1} &= s_n \end{aligned}$$

where the unknown coefficients $b_{i,j} = a_j \cdot w_i^{-1}$, $1 \leq i \leq n$, $0 \leq j \leq t-1$, are all distinct from each other. Given the public parameters $t_j = \alpha^{k_j} \in B$, we have that $k_j \in K$ is accordingly protected due to the Discrete Logarithm Problem.

The equation system is composed of $(t \cdot n)$ unknown coefficients which makes it is infeasible to solve since there are too many unknowns. This has two important

security implications: 1) Since the equation system is thus prohibited from being solved, the secret polynomial coefficients are successfully being protected. This prohibits bypassing of the threshold requirement and thus, Security Requirement 1 is preserved. 2) Since the coefficients cannot be deduced, it is prohibited that any user coalition can illegitimately establish new user shares or reestablish existing secret user shares, thus complying with Security Requirement 2. Though any user coalition may compute α^{a_j}, $0 \leq j \leq t-1$, the exponent a_j is protected due to the Discrete Logarithm Problem.

Security Requirement 2. Protection of the user shares. In agreement with analysis of Security Requirement 1, disclosure of the secret coefficients $(a_0, \ldots, a_{t-1})$ and the secret factor w_i is prevented. Since new user shares can only be established by means of these coefficients, it is prevented that any user coalition can illegitimately establish new user shares or reestablish existing secret user shares.

In the decryption phase, each participant $P_i \in T$ computes Y_i by applying his or her secret share s_i as an exponent modulo p. Obtaining the secret user share from Y_i is equivalent to solving the Discrete Logarithm Problem.

Security Requirement 2 is hence preserved.

Security Requirement 3. Protection of encrypted data. Given $z_j = t_j^r \bmod p$, $j \in \{1, \ldots, m\}$, it is computationally infeasible to obtain the secret r due to the Discrete Logarithm Problem. This corresponds to the cryptographic trapdoor of the ElGamal cryptosystem. Thus, the difficulty of obtaining the secret encryption factor $\alpha^{a_0 \cdot r}$ is equivalent to breaking the ElGamal cryptosystem.

3 Conclusion

In this paper, we have presented the collusion problem related to threshold-oriented cryptosystems. The underlying threshold secret sharing scheme of threshold cryptosystems does not prohibit a minimum number of colluding participants to compute the shared secret or any of the polynomial coefficients. Obtaining the shared secret enables a user to bypass the threshold requirement and to carry out threshold-oriented cryptographic computations on an individual basis. Obtaining all the secret polynomial coefficients would moreover enable establishment new user shares.

In this paper, we have proposed a method applied to the threshold decryption scheme of Frankel and Desmedt that prohibits deduction of any of the secret polynomial coefficients of the underlying secret polynomial, and thereby preventing violation of the threshold security requirement.

References

[1] Y. Desmedt, Y. Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307–315, Springer-Verlag, 1990.

[2] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, Vol. 547, pp. 522–526, Springer-Verlag, 1991.

[3] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner. LNCS, Vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[4] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings. Vol. 141, No. 5, pp. 307-313, 1994.

[5] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders, Eurocrypt 1994, pp. 194–204.

[6] C. Li, J. Pieprzyk. Conference Key Agreement from Secret Sharing. ACISP '99, Springer-Verlag, pp. 64–76, 1999.

[7] J. Pieprzyk, C. H. Li. Multiparty key agreement protocols. IEE Proceedings, Computer and Digital Techniques, Vol. 147, No. 4, pp. 229–236, 2000.

[8] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612–613, 1979.

[9] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469–472, 1985.

Paper E

Collusion-Resistant Threshold Cryptosystems

Sigurd Eskeland and Vladimir Oleshchuk

Collusion-Resistant Threshold Cryptosystems

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract

Threshold-oriented cryptosystems require at least t arbitrary participants of a group of n (where $t \leq n$) to carry out computation of a cryptographic function, e.g., threshold decryption or threshold signature computation. Such cryptosystems commonly incorporate the secret sharing scheme of Shamir which is based on the secrecy of a polynomial. Given t user shares, Lagrange interpolation enables disclosure of the secret polynomial coefficients, whereof one constitutes a secret shared key. Knowledge of the shared key enable bypassing of the threshold security requirement, so that such computations can be carried out on an *individual* basis. The threshold security requirement would thus be violated, and it cannot be guaranteed that allegedly subsequent threshold computations have actually been carried out by means of collaboration and not by a single individual. Moreover, disclosure of the polynomial coefficients enables establishment of new user shares. We refer to this as the collusion problem. This could be a serious problem when the threshold is low, for example 2 or 3.

In this paper, we present a method that prevents computation of any of the secret coefficients of the polynomial of the underlying Shamir secret sharing scheme, thus preventing the collusion problem in threshold cryptosystems.

1 Introduction

In the threshold secret sharing scheme of Shamir [12], a secret key is shared (or split) among n members of a group so that each is allocated a unique share of the secret. Each user share is computed from a secret polynomial, where the secret key would normally be the least-order coefficient of the secret polynomial. The shared secret can then later on be recovered from at least t arbitrary user shares computed from the same polynomial, while less than t shares reveal no information about the shared secret. The threshold property provides flexibility since it enables an arbitrary composed subgroup of t out of n participants to collaboratively carry out computation of a given cryptographic function. In

contrast, assuming that a predefined, fixed set of specific participants are to be required to carry out a cryptographic action, would be a major disadvantage, particularly since such an action would be impeded in any case of absence of one or more of the pertaining participants.

Threshold cryptosystems are generally based on the threshold-secret sharing scheme of Shamir, which provide the security requirement that computations can only be carried out by group consensus and hence collaboration of a minimum number of associated parties. We refer to this security requirement as the threshold requirement. An essential security objective of threshold cryptosystems is therefore to prevent that individuals can carry out threshold-computations individually.

This security property is desirable in scenarios where some sort of separation of duty is required. A practical example of this could be access to a bank vault. It is not desirable that one person alone would control the key to the vault due to the risk of fraud, robbery and extortion. Rather, the vault should only be unlockable by agreement and cooperation of at least t bank employees. However, since employees could be prohibited from attending work due to sickness, death, or any other reason, a practical and flexible vault locking mechanism could be based on the threshold property, and open as a function of the secret keys of t arbitrary employees of certain clearance.

Threshold cryptosystems based on the Shamir secret sharing scheme allow the secret polynomial coefficients, including the shared secret, to be restored by Lagrange interpolation given t arbitrary user shares. Disclosure of the shared secret would allow anyone holding the shared secret to later on carry out threshold-computations on an *individual* basis. The threshold security requirement would thus be violated. Consequently, although t participants have to collude in the first place, this means that it cannot be guaranteed that subsequent alleged threshold computations have actually been carried out by means of collaboration and not by a single individual.

Disclosure of all the secret polynomial coefficients would moreover enable illegitimate establishment of any user share $s_i = f(i)$ for any i, where f is the pertaining polynomial. We refer to this as the collusion problem. The smaller the threshold is, for instance 2 or 3, the more imminent the collusion problem may be.

The collusion problem regarding threshold cryptosystems was first addressed in [3], where a collusion-resistant threshold decryption scheme was proposed. Its computational overhead for encryption is $\log_2 n$ additional exponentiations to the underlying threshold cryptosystem, where n denotes the total number of participants in the group.

In this paper, we propose a collusion-resistant threshold decryption scheme, whose novelty is the use of rational functions for providing collusion resistance to any number of colluding participants. The proposed scheme has a very small computational overhead of only one additional exponentiation to the underlying threshold cryptosystem.

2 Background

Typical threshold-oriented applications are threshold decryption and threshold signatures. Threshold decryption cryptosystems enforce that a minimum number of t arbitrary participants of a group of n participants are required to collaboratively carry out decryptions pertaining to their group. When a group is represented by a public key, outsiders can confidentially address the group. Only by collaborating in such a way that the active group members are each contributing with their partial computational results, the encrypted message can be decrypted. See [5–7] for relevant threshold decryption cryptosystems. A discussion of diverse theoretical issues related to threshold cryptography is given in [4]. Likewise, threshold signature schemes (e.g., [8, 9]), require a minimum coalition to compute digital signatures due to the threshold requirement. Note that threshold signatures do not reveal the identity of the actual signing group members.[2] Another threshold-oriented security application is conference key establishment [10, 11].

2.1 Shamir secret sharing

In the (t, n) threshold secret sharing scheme of Shamir [12], a secret number is split into n secret user shares. The secret can only be reconstructed by means of an arbitrarily composed subset of minimum t user shares, while less than t shares reveal no information about the shared secret. A dealer or trusted authority (TA) sets up the scheme by generating a secret polynomial of order $(t-1)$:

$$f(x) = \sum_{j=0}^{t-1} a_j \cdot x^j$$

Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ denote a group of n participants. For a given $\mathcal{U}$, the TA computes for each $P_i \in \mathcal{U}$ a secret user share s_i according to

$$s_i = f(i) \pmod{q}$$

where q is a large prime. Normally, the shared secret is defined as $f(0) = a_0$, but it can in general be f for any $x \in \mathbb{Z}_q$.

By means of at least t user shares, all polynomial coefficients $(a_0, \ldots, a_{t-1})$ can be restored by Lagrange interpolation (see Appendix A for more details), or by solving the linear equation system

$$\begin{pmatrix} 1 & x_1 & \ldots & x_1^{t-1} \\ 1 & x_2 & \ldots & x_2^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_t & \ldots & x_t^{t-1} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{t-1} \end{pmatrix} = \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{t-1} \end{pmatrix}$$

corresponding to t user shares.

[2] Note the distinction from group signatures (that are not threshold-based), where a single member signs on behalf of a group in such a way that the identity of the signer cannot be determined, only that the signer is member of a specific group.

2.2 Threshold decryption

In this subsection, we illustrate the threshold decryption scheme proposed by Desmedt and Frankel [5]. This scheme is equivalent to the ElGamal public key cryptosystem [13], where the private key is split by means of Shamir secret sharing [12].

A trusted authority (TA) sets up the scheme by computing the secret user shares for a given user group $\mathcal{U}$ according to Shamir secret sharing scheme. The TA selects two large public primes p and q, where $p = 2 \cdot q + 1$. The TA moreover selects a generator α to $\mathbb{Z}_q$. The team is represented by the public key $y = \alpha^{-a_0} \pmod p$, where $a_0 = f(0)$ represents the private, shared group key. The encryption and decryption functions are defined as follows:

Encryption. By means of the public key y, an encrypting entity (or sender) encrypts a message m according to the ElGamal public key cryptosystem. The sender generates a random secret number r from $\mathbb{Z}_q$, and computes the cryptogram (c, R), where

$$c = m \cdot y^r \pmod p \quad \text{and} \quad R = \alpha^r \pmod p$$

and sends (c, R) to $\mathcal{U}$.

Decryption. To decrypt (c, R), the partial computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required, i.e., $|T| \geq t$. Each $P_i \in T$ computes and sends

$$Y_i = R^{s_i} \pmod p$$

confidentially, i.e., via secure channels, to the other participants of T. Let $I_T = \{i \mid P_i \in T\}$. By means of the partial computations, the plaintext is restored by using Lagrange interpolation on the exponents:

$$m = c \cdot \prod_{j \in I_T} Y_j^{b_j} = c \cdot R^{\sum_{j \in I_T} s_j \cdot b_j} = c \cdot R^{a_0} \pmod p$$

where $b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{j}{j-i} \pmod q$.

2.3 Bypassing the threshold requirement

As noted above, one of the main purposes of threshold-oriented cryptographic schemes is to enforce a consensus requirement as a condition for computation of a cryptographic function. However, a user coalition $T \subseteq \mathcal{U}$ can compute the shared private group key a_0 by using Lagrange interpolation, as $a_0 = \sum_{j \in I_T} s_j \, b_j \pmod q$. Disclosing a_0 would allow any user holding it to individually decrypt of all preceding and future cryptograms pertaining to the team given (c, R):

$$m = c \cdot R^{a_0} \pmod p$$

Thus, the threshold requirement is violated. The lower t is, for instance 2 or 3, the more imminent could this be.

In the rest of this subsection, we will describe an inefficient collusion-resistant variant of the threshold decryption scheme. A straight-forward way to make the threshold decryption scheme collusion-resistant could be to hide $f(i)$ by multiplication of a secret factor k_i that is unique for each user $P_i \in \mathcal{U}$. The user share for each $P_i \in \mathcal{U}$ would be $s_i^* = k_i \cdot f(i) \bmod q$. This would prevent that Lagrange interpolation can be computed by such user shares due to that q is a large prime, and the polynomial coefficients of $f(x)$ would therefore be protected.

However, Lagrange interpolation could be achieved by exponential computation given the public numbers $y_i = \alpha^{(k_i^{-1})} \bmod p$ for each $P_i \in \mathcal{U}$. Due to the Discrete Logarithm Problem, it would be computationally infeasible to compute the secret k_i given y_i.

The encrypting entity (the sender) computes $z_i = y_i^r \bmod p$ for each user $P_i \in \mathcal{U}$. Each $P_i \in T$ would subsequently compute $v_i = z_i^{s_i^*} = R^{k_i^{-1}(k_i \cdot f(i))} = R^{f(i)} \bmod p$. These computational results would then be used for subsequent Lagrange interpolation on the exponents according to Section 2.2. Although $\alpha^{f(x)} \bmod p$ can be computed for any x, it would be computationally infeasible to deduce $f(x)$ due to the Discrete Logarithm Problem. However, this method is relatively inefficient, since it requires $n = |\mathcal{U}|$ exponentiations on the sender side and transferral of n additional numbers.

3 Preliminaries

In this section, we present the security requirements for the proposed scheme, and some mathematical observations relevant for understanding it.

3.1 Security requirements

In order to provide collusion-resistance to threshold cryptosystems based on the Shamir secret sharing scheme, the following security requirements must be satisfied:

Security Requirement 1. *Collusion-resistant security of polynomial coefficients.* It must be computationally infeasible to restore any polynomial coefficients of the polynomial constituting the underlying Shamir scheme, given any number of user shares.

Security Requirement 2. *Collusion-resistant security of user shares.* It must be computationally infeasible to deduce existing or compute new user shares given any number of user shares.

Security Requirement 3. *Cryptographic security.* The cryptographic security must be in agreement with the underlying threshold cryptosystem.

The second security requirement is tightly connected with Security Requirement 1 since the user shares are computed as a function of the secret polynomial, but it should nevertheless be explicitly stated. The last security requirement pertains to the security of the actual underlying cryptosystem.

3.2 Relevant mathematical observations

Let us consider the following rational function consisting of two first order polynomials

$$s_i = \frac{f(i)}{g(i)} = \frac{a_0 + a_1 \cdot i}{d_0 + d_1 \cdot i} \quad (1 \leq i \leq n)$$

where we assume that the polynomial coefficients of f and g are unknown. In order to find (a_0, a_1, d_0, d_1), we can write this as an equation $a_0 + i\,a_1 = s_i\,d_0 + i\,s_i\,d_1 \Leftrightarrow a_0 + i\,a_1 - s_i\,d_0 - i\,s_i\,d_1 = 0$.

For instance, given 4 user shares $(s_1, \ldots, s_4)$ we can set up the equivalent system of linear equations:

$$\begin{bmatrix} 1 & 1 & -s_1 & -s_1 \\ 1 & 2 & -s_2 & -2\,s_2 \\ 1 & 3 & -s_3 & -3\,s_3 \\ 1 & 4 & -s_4 & -4\,s_4 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ d_0 \\ d_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Let $ord(f) \geq 1$ and let $ord(g) = 1$, where $ord(f)$ denotes the polynomial order of f. Let the polynomial order of f define the threshold t, i.e, $t = ord(f) + 1$. Generalizing with respect to a threshold t is equivalent to the following linear system requiring $t + 2$ user shares $(s_1, \ldots, s_{t+2})$:

$$\begin{bmatrix} 1 & \ldots & 1 & -s_1 & -s_1 \\ 1 & \ldots & 2^{t-1} & -s_2 & -2\,s_2 \\ & \vdots & & & \\ 1 & \ldots & (t+1)^{t-1} & -s_{t+1} & -(t+1)\cdot s_{t+1} \\ 1 & \ldots & (t+2)^{t-1} & -s_{t+2} & -(t+2)\cdot s_{t+2} \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ \vdots \\ a_{t-1} \\ d_0 \\ d_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

Such linear equation systems where the constant terms constitute a zero-vector are known as homogenous linear systems. Homogenous linear systems are equivalent to linear equation systems where the number of equations is lower than the number of unknowns (see Appendix B for an example), and have therefore in general either a zero solution or infinitely many solutions (e.g., [1, p. 377]). Thus, if all polynomial coefficients are other than zero, we are left with an infinite number of solutions. The equation system is therefore underdefined, and cannot be solved. Hence, the secret coefficients $(a_0, \ldots, a_{t-1}, d_0, d_1)$ are prevented from disclosure.

4 Collusion-resistant threshold decryption

In this section, we present the efficient collusion-resistant threshold decryption scheme.

Initializations. Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ denote a team of n participants. A trusted authority (TA) selects two large public primes p and q such that $p = 2 \cdot q + 1$, and a generator α to $\mathbb{Z}_q$ (e.g., [2, p. 30]). The TA defines the threshold t, and randomly generates two distinct secret polynomials:

$$f(x) = \sum_{j=0}^{t-1} a_j \cdot x^j \quad \text{and} \quad g(x) = d_0 + d_1 \cdot x$$

The TA computes a secret user share

$$s_i = \frac{f(i)}{g(i)} \pmod{q}$$

for each $P_i \in \mathcal{U}$. Furthermore, the TA computes

$$y = \alpha^{-a_0} \pmod{p}, \quad z_0 = \alpha^{d_0} \pmod{p} \quad \text{and} \quad z_1 = \alpha^{d_1} \pmod{p}$$

where $a_0 = f(0)$ is the shared secret.

Encryption. In order to encrypt the message m, the encrypting entity (the sender) generates a random secret number r from $\mathbb{Z}_q$ and computes

$$c = m \cdot y^r \pmod{p} \quad \text{and} \quad W = \left\{ w_j = z_j^r \bmod p \,|\, j \in \{0, 1\} \right\}$$

and sends (c, W) to $\mathcal{U}$. Note that there is only one additional exponentiation compared to the ElGamal public key cryptosystem.

Decryption. To decrypt the cryptogram, the partial computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required. Each $P_i \in T$ computes

$$v_i = \left(w_0 \cdot w_1^i\right)^{s_i} = \left(\alpha^{r \cdot g(i)}\right)^{\frac{f(i)}{g(i)}} = \alpha^{r \cdot f(i)} \pmod{p}$$

and confidentially shares the result with the the other participants of T. Let $I_T = \{i \,|\, P_i \in T\}$. Using the Lagrange interpolation method on the exponents, the secret encryption factor is restored according to

$$Y = \prod_{j \in I_T} v_j^{b_j} = (\alpha^r)^{\sum_{j \in I_T} f(j) \cdot b_j} = \alpha^{r \cdot a_0} \pmod{p}$$

where $b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{j}{j-i} \pmod{q}$, and subsequently the plaintext

$$m = c \cdot Y \pmod{p}$$

in agreement with the ElGamal public key cryptosystem [13].

4.1 Security analysis

In this section, we provide a security analysis showing that the presented scheme is secure in agreement with the security requirements presented in Section 3.1.

Security Requirement 1. Collusion-resistant security of polynomial coefficients. The secret polynomial coefficients $(a_0, \ldots, a_{t-1}, d_0, d_1)$ provide the basis for computing secret user shares s_i, $P_i \in \mathcal{U}$. The public parameters (y, z_0, z_1) are computed based on the secret polynomial coefficients (a_0, d_0, d_1), respectively. These secret coefficients are protected in agreement with the following:

- Using the secret user shares as exponents, the value $R^{f(i)} \pmod p$ can be computed for any $P_i \in \mathcal{U}$:

$$R^{f(i)} = \left(w_0 \cdot w_1^i\right)^{s_i} = \left(R^{g(i)}\right)^{\frac{f(i)}{g(i)}} = v_i \pmod p$$

  Computing $f(i)$ from $R^{f(i)} \bmod p$ is equivalent of solving the Discrete Logarithm Problem. Since it is prevented that $f(i)$ for any $P_i \in \mathcal{U}$ can be disclosed, it is therefore prevented that the secret polynomial $(a_0, \ldots, a_{t-1})$ can be disclosed.
- Regarding the public parameters $(y = \alpha^{a_0} \bmod p, z_i = \alpha^{d_i} \bmod p, i \in \{0, 1\})$, the secret (a_0, d_0, d_1) are protected due to the Discrete Logarithm Problem.

Since the number of colluding participants does not affect the hardness of the Discrete Logarithm Problem, this leaves the following problem. According to Section 3, $t + 2$ or more user shares constitute a homogenous linear equation system. Solving such an equation system results in a linear equation system where the number of equations are less than the number of unknown coefficients, which results in an infinite number of solutions. The equation system is therefore underdefined, and cannot be solved given any number of user shares.

Computation of any of the secret coefficients $(a_0, \ldots, a_{t-1}, d_0, d_1)$ is is therefore prevented regardless of the number of colluding participants. Security Requirement 1 is therefore preserved.

Security Requirement 2. Collusion-resistant security of user shares. In agreement with the analysis of Security Requirement 1, disclosure of the secret coefficients $(a_0, \ldots, a_{t-1}, d_0, d_1)$ is prevented. Since new user shares can only be established by means of there coefficients, it is prevented that any user coalition can illegitimately establish new user shares or reestablish existing secret user shares.

In the decryption phase, each participant $P_i \in T$ computes the partial encryption factor v_i by applying his or her secret share s_i as an exponent modulo p. Computing s_i given v_i is equivalent to solving the Discrete Logarithm Problem, and is therefore computationally infeasible. Since the number of colluding participants do not affect the hardness of the Discrete Logarithm Problem, collusion-oriented computation of user shares is is therefore prevented regardless of the number of colluding participants. Thus, Security Requirement 2 is preserved.

Security Requirement 3. Cryptographic security. This security property corresponds to security of the underlying threshold cryptosystem. The security is based on the difficulty of computing the secret encryption factor $Y = y^r = \alpha^{r \cdot a_0} \bmod p$, where a_0 and r are secret, and W is public. Computing r from W is equivalent to solving the Discrete Logarithm Problem. Computing the secret encryption factor Y is equivalent to breaking corresponding the ElGamal cryptosystem, which is the basis for the threshold cryptosystem. Thus, Security Requirement 3 is preserved.

5 Conclusion

In this paper, we have shown that the underlying Shamir secret sharing scheme of threshold-cryptosystems does not prevent a minimum number of colluding participants to compute the shared secret key or any of the polynomial coefficients. By using the shared secret key, a user can bypass the threshold requirement by carrying out threshold-oriented cryptographic computations on an individual basis. We have presented a collusion-resistant threshold decryption scheme that prevents computation of any of the secret polynomial coefficients of the underlying secret polynomials. violation of the threshold security requirement is thereby prevented. Future work could be to develop a scheme for collusion-resistance threshold signatures.

Appendix A.

Lagrange interpolation. Let $I_T = \{i \mid P_i \in T\}$. The Lagrange polynomial is given by

$$f(x) = \sum_{i \in I_T} s_i \, b_i(x) \quad \text{where} \quad b_i(x) = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x - j}{i - j} = \sum_{j=0}^{t-1} c_{i,j} \, x^j$$

Thus, by multiplying out the factors of $b_i(x)$, we obtain the Lagrange basis coefficients $c_{i,j}$ whereof we can compute the polynomial coefficients a_j of $f(x)$:

$$f(x) = \sum_{i \in I_T} s_i \, b_i(x) = \sum_{i \in I_T} s_i \, (\sum_{j=0}^{t-1} c_{i,j} \, x^j) = \sum_{j=0}^{t-1} (\sum_{i \in I_T} s_i \, c_{i,j}) x^j = \sum_{j=0}^{t-1} a_j \, x^j$$

Appendix B.

In this section, we present a small example of the proposed scheme represented by the equivalent linear system that it constitutes. The case demonstrates how

the corresponding equation system can be reduced with respect to the coefficients and how effectively the coefficients are protected. For this example, we assume that all numbers are in $\mathbb{R}$.

Let $s_x = \frac{a_0+a_1\,x}{d_0+d_1\,x}$ where $a_0 = 3$, $a_1 = 7$, $d_0 = 5$, $d_1 = 4$ and whereof we get four user shares $s_1 = \frac{10}{9}$, $s_2 = \frac{17}{13}$, $s_3 = \frac{24}{17}$, $s_4 = \frac{31}{21}$. We represent this as linear equation system in augmented matrix form, and solve it according to Gauss-Jordan elimination[3](see e.g., [1, p. 362]):

$$\left[\begin{array}{cccc|c} 1 & 1 & -s_1 & -s_1 & 0 \\ 1 & 2 & -s_2 & -2\,s_2 & 0 \\ 1 & 3 & -s_3 & -3\,s_3 & 0 \\ 1 & 4 & -s_4 & -4\,s_4 & 0 \end{array}\right] = \left[\begin{array}{cccc|c} 1 & 1 & -\frac{10}{9} & -\frac{10}{9} & 0 \\ 1 & 2 & -\frac{17}{13} & -\frac{34}{13} & 0 \\ 1 & 3 & -\frac{24}{17} & -\frac{72}{17} & 0 \\ 1 & 4 & -\frac{31}{21} & -\frac{124}{21} & 0 \end{array}\right] \begin{array}{c} M_2(13) \\ M_3(17) \\ M_4(21) \\ \longrightarrow \end{array}$$

$$\left[\begin{array}{cccc|c} 1 & 1 & -\frac{10}{9} & -\frac{10}{9} & 0 \\ 13 & 26 & -17 & -34 & 0 \\ 17 & 51 & -24 & -72 & 0 \\ 21 & 84 & -31 & -124 & 0 \end{array}\right] \begin{array}{c} A_{1,2}(-13) \\ A_{1,3}(-17) \\ A_{1,4}(-21) \\ \longrightarrow \end{array} \left[\begin{array}{cccc|c} 1 & 1 & -\frac{10}{9} & -\frac{10}{9} & 0 \\ 0 & 13 & -\frac{23}{9} & -\frac{176}{9} & 0 \\ 0 & 34 & -\frac{46}{9} & -\frac{478}{9} & 0 \\ 0 & 63 & -\frac{23}{3} & -\frac{302}{3} & 0 \end{array}\right] \begin{array}{c} M_2\left(\frac{1}{13}\right) \\ M_3(9) \\ M_4(3) \\ \longrightarrow \end{array}$$

$$\left[\begin{array}{cccc|c} 1 & 1 & -\frac{10}{9} & -\frac{10}{9} & 0 \\ 0 & 1 & -\frac{23}{117} & -\frac{176}{117} & 0 \\ 0 & 306 & -46 & -478 & 0 \\ 0 & 189 & -23 & -302 & 0 \end{array}\right] \begin{array}{c} A_{2,1}(-1) \\ A_{2,3}(-306) \\ A_{2,4}(-189) \\ \longrightarrow \end{array} \left[\begin{array}{cccc|c} 1 & 0 & -\frac{107}{117} & -\frac{46}{117} & 0 \\ 0 & 1 & -\frac{23}{117} & -\frac{176}{117} & 0 \\ 0 & 0 & \frac{184}{13} & -\frac{230}{13} & 0 \\ 0 & 0 & \frac{184}{13} & -\frac{230}{13} & 0 \end{array}\right] \begin{array}{c} M_3\left(\frac{13}{184}\right) \\ M_4\left(\frac{13}{184}\right) \\ \longrightarrow \end{array}$$

$$\left[\begin{array}{cccc|c} 1 & 0 & -\frac{107}{117} & -\frac{46}{117} & 0 \\ 0 & 1 & -\frac{23}{117} & -\frac{176}{117} & 0 \\ 0 & 0 & 1 & -\frac{230}{184} & 0 \\ 0 & 0 & 1 & -\frac{230}{184} & 0 \end{array}\right] \begin{array}{c} A_{3,1}\left(\frac{107}{117}\right) \\ A_{3,2}\left(\frac{23}{117}\right) \\ A_{3,4}(-1) \\ \longrightarrow \end{array} \left[\begin{array}{cccc|c} 1 & 0 & 0 & -\frac{3}{4} & 0 \\ 0 & 1 & 0 & -\frac{7}{4} & 0 \\ 0 & 0 & 1 & -\frac{230}{184} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

The last matrix corresponds to

$$\begin{aligned} a_0 \quad\quad\quad\quad -\tfrac{3}{4}\,d_1 &= 0 \\ a_1 \quad\quad\quad\quad -\tfrac{7}{4}\,d_1 &= 0 \\ d_0 \quad -\tfrac{230}{184}\,d_1 &= 0 \end{aligned}$$

Since we have more unknowns than equations, the system has an infinite number of solutions, effectively hiding the actual values of the coefficients.

[3]Notation. 1) $M_i(c)$ means multiply the ith row by c. 2) $A_{i,j}(c)$ means multiply the ith row by c and add it to the jth row.

References

[1] S. Grossmann, W. Derrick. Advanced Engineering Mathematics. ISBN 0-06-042534-2. Harper-Collins Publishers, 1988.

[2] J. Pieprzyk, T. Hardjono, J. Severry. Fundamentals of computer security. ISBN 3-540-43101-1, Springer-Verlag, 2003.

[3] S. Eskeland, V. Oleshchuk. Collusion-resistant threshold decryption. Proc. of fourth IASTED conference on Communication, Network and Information Security (CNIS), pp. 12–15, 2007.

[4] Y. Desmedt. Some Recent Research Aspects of Threshold Cryptography. Information Security, LNCS 1396, pp. 158–173. Springer-Verlag, 1997.

[5] Y. Desmedt, Y. Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307–315, Springer-Verlag, 1990.

[6] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, Vol. 547, pp. 522–526, Spring-Verlag, 1991.

[7] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner. LNCS, Vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[8] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings. Vol. 141, No. 5, pp. 307-313, 1994.

[9] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders, Eurocrypt 1994, pp. 194–204.

[10] C. Li, J. Pieprzyk. Conference Key Agreement from Secret Sharing. ACISP '99, Springer-Verlag, pp. 64–76, 1999.

[11] J. Pieprzyk, C. H. Li. Multiparty key agreement protocols. IEE Proceedings, Computer and Digital Techniques, Vol. 147, No. 4, pp. 229–236, 2000.

[12] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612–613, 1979.

[13] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469–472, 1985.

Paper F

Anonymity Preserving Authorization Granting In Medical Information Networks

Sigurd Eskeland and Vladimir Oleshchuk

Anonymity Preserving Authorization Granting In Medical Information Networks

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract

Due to the sensitivity of personal medical information, this paper addresses the need of hiding patient identities — in contrast to only keeping their medical data confidential. Thus, it could in some cases be desirable that personal and meaningful patient identity information like names, addresses, personal identity numbers, etc., would not to be linked to disclosed electronic patient records (EPR). To achieve this, we propose a scheme that enables patients to anonymously grant medical teams authorization to access their EPRs without revealing their true identities to the medical practitioners. An essential benefit is that it enables patients to exert control over their own medical data. A security evaluation is included.

1 Introduction

With the emerge of information technology in health care, there has been extensive focus on the security issues of electronic patient records (EPR) in medical environments. These issues include how to ensure that *only* legitimate personnel can access no more than the required electronic patient records in order to provide medical care to the concerning patients, and moreover, how to ensure that medical information is preserved and managed confidentially.

Although medical patient data remain confidential, it may be cases when it is desirable that the identities of patients remain confidential as well, even after disclosure of patient data. Concerning personal information about AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, sexual divergencies, genetic predispositions to diseases, information about toxic addictions, and so on [1], it is likely that some patients wish to remain anonymous. Thus, meaningful identity information such as names, birthdays, personal identity numbers, addresses etc., must not be linked to corresponding disclosed EPRs. Likewise, for purposes such as medical research, disclosed medical data should not be linked to the concerning patients. Access control should nevertheless be maintained properly so that EPRs are not accessible to other medical

employees than legitimate medical teams providing medical care to the concerning patients.

The problem boils down to how to link patients with their respective EPR without revealing personal identity information. In this paper, we propose a cryptographic solution that enables patients to grant medical teams authorization to access their corresponding EPRs in such a way that the real identity of the patients are not disclosed. Thus, patients remain anonymous, and patients, teams and EPR security server can nevertheless authenticate each other.

Patients granting authorization implicitly state consent and exert control over their own medical data by controlling who can access their corresponding EPRs. Moreover, by removing personal identity information from EPRs, privacy is preserved when medical data is disclosed for medical research.

In the case that a patient is unconscious and therefore unable to consent access to the EPR, the security system can include an emergency mode where one, or preferably a minimum coalition of two (or three) security administrators, approve a medical team access on behalf of the concerning patient.

2 Previous work

It has been previously proposed to remove personal identity information (names, addresses, phone numbers, etc.) to provide anonymity in medical environments [2]. However, this may not rule out the possibility that it may be possible to correlate anonymous medical data to the corresponding individuals. Sweeney [3] proposes methods of substitution of data to prohibit such correlations. In order to enforce access control, explicit identifiers are required to link data records. It is assumed in this paper that the medical data themselves contain no explicit references to the patient.

In [4, 5], the authors propose anonymization of patient data based on encrypted anonymous identifiers or pseudonyms. Patients can consent to disclose their medical data by supplying their pseudonyms, but have, however, no technological enforcement about who is to access the medical data. Another problem is the staticness of the pseudonyms, and that there is no challenge-response mechanisms ensuring online certification of requests. Thus, an adversary can obtain an encrypted anonymous identifier by eavesdropping. By replaying this, the security server will not be able to distinguish whether a request originated from the patient or not. Moreover, there is no mechanism that ensures the security server that no other than legitimate medical professionals, and not an adversary, is targeted to access the EPR.

3 Anonymous authentication and authorization

3.1 A framework for patient anonymity

In medical networks, access to EPRs includes known identities of patients. This could be meaningful public identifiers like names and/or personal security numbers. Such identifiers would then be known to the parties involved in the authentication and authorization process. Consequently, the medical personnel that provides medical care depends on knowing such identities in order to provide the right care to the right patients.

In circumstances where patients require confidentiality and want to remain anonymous, consequently, it is essential that names and social security numbers (SSN) cannot be associated with their corresponding EPRs. This means that it is not desirable to have fully trusted administrators who can obtain such relations. Moreover, the EPR server must not link (or contain a table that links) patient names/SSNs with corresponding EPRs. In contrary, in this paper we propose an approach where the EPR server associates each EPR entry with a specially generated anonymous EPR identifier (AEID). The AEID is partly based on a secret long-term key held by the corresponding patient. This secrecy of this key prohibits any other party including administrative personnel from obtaining associations between patients and their EPRs. Basically, given a patient name or SSN, it must be infeasible to obtain the corresponding EPR entry and the EPR of the patient without knowing the key.

A semi-trusted administrator (STA) may be needed to assign available and appropriate medical professionals for treatment of incoming patients. It may be necessary that the STA knows names and SSN of hospitalized patients, but there is no need for this party to access EPRs. In our approach, the STA, being an administrative and coordinating entity having knowledge of names and SSN of hospitalized patients, has not a role and authority to assign authorizations on behalf of patients for medical professionals access their EPRs, nor to be able to associate names/SSNs of patients with their medical records.

The EPR server contains a table that links the anonymous EPR identifiers (AEID) and the associated EPR entries of each patient. This table does not provide any relationships between patient names and AEIDs. Furthermore, none of the involved parties, including patients, know or can obtain the association between an AEID and actual patient identities.

To ensure patient anonymity, medical personnel is not given any name/SSN information of patients, but patients are instead referenced by means of a temporary anonymous patient identifier (TAPI). The duration of the hospitalization reflects the lifetime of a TAPI, and a new unique TAPI must be established for each hospitalization.

The correspondence between TAPI and AEID can only be obtained by the EPR server. Thus, if STA has obtained access to the AEID/EPR entry table where each record is referenced by AEID, the STA has no way to determine which

Entity	*Identifier*
STA	$TAPI_i$, SSN_i
S	$TAPI_i$, $AEID_i$
P_i	$TAPI_i$, SSN_i
T	$TAPI_i$

Table 1: Relationship between identifiers known by which entities

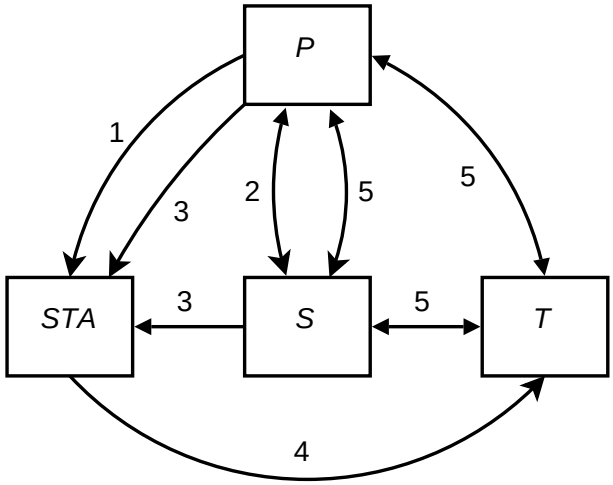

Figure 1: The hospitalization process

record that is associated by any TAPI. Table 1 shows which identifiers known by the involved entities, and the association of the identifiers. The entity STA denotes a semi-trusted administrator, S denotes the EPR server, P_i denotes a patient and T a medical team.

The hospitalization process of a patient is summarized in Fig. 1 and as follows:

1. First at hospitalization, the patient may identify himself by name and SSN to the STA for administrative purposes like billing.
2. The patient anonymously certifies himself to the EPR server by means of his or her secret key, and generates TAPI.
3. The patient supplies the value of TAPI to STA. The EPR server acknowledges by supplying TAPI to STA. The STA compares the to received values – match indicates that the patient is legitimate.
4. STA assigns a medical team for the patient referenced by TAPI.
5. The team, EPR server and patient (by means of TAPI) authenticates each other. By completing the authentication, the patient is granting the medical team authorization to access his EPR.

An important property here is that the patient's involvement is required in order to obtain access to his or her EPR. Since only the corresponding patient holds

the private key, this provides the patient the authority to grant arbitrary teams of medical professions access to his EPR.

3.2 Security properties and requirements

In the problem setting considered in this paper, there are three active entities: The medical team T, the granting patient P, and the EPR security server S. We have the following security requirements:

Authentication: The involved parties must mutually authenticate each other.

Authorization: Only P can authorize T to access his or her EPR.

Anonymity: The identity of P must be hidden from T and S.

Unlinkability: It is infeasible to deduce that various TAPIs may refer to the same P.

The first security requirement embraces the following aspects:

- T and S must authenticate themselves to P, preventing that an adversary masquerading as T or S could illegitimately obtain authorization
- P anonymously authenticates himself to S to prohibit that an adversary may successfully masquerade as P. Likewise, T must authenticate themselves to S.
- S must authenticate itself to T and thereby confirming whether P is a valid patient and whether T is granted authorization. Recall that the real identity of P is hidden to T, and that P is referenced by TAPI.

Successful and completed execution of the proposed anonymity-preserving authentication and authorization protocol provides granted authorization. Thus, the patient must authenticate the medical team and security server first, since the patient is the granting entity. Subsequently, the EPR server authenticates the patient and team.

The third security requirement means that the given patient is not referenced by his name/SSN, but rather a temporary TAPI. It should be infeasible to obtain associations between any patients and their TAPIs. This is related to the fourth which is to ensure unlinkability between TAPI and AEID of the same patient for all other than patients and EPR server. It is likewise infeasible to obtain relationships between any names/SSNs and AEIDs.

Assuming that the network is not secure, it can be assumed that all messages exchanged over the network can be eavesdropped by an adversary. Thus, the protocol must resist passive attacks like eavesdropping, and active attacks by manipulation and substitution of messages where the adversary may be masquerading as a legitimate entity.

Public keys should be certifiable, for example represented by digital certificates or be identity-based so that substitution attacks of public keys will be detected.

Successful replacement of public keys would consequently break the security of the protocol (and any other protocol).

Since all entities hold private keys that are actively involved in the authentication and authorization process, these should be held and stored securely, for example in secure tamper-free devices. Computations should be done in these devices so that the personal keys are never disclosed. Personal smartcards are a possible solution to satisfy these requirements. However, this part is outside the scope of this paper.

4 The anonymity preserving authorization protocol

The objective of this protocol is to enable patients to anonymously grant medical teams authorization to access their EPRs without revealing identities. The protocol is three-fold. Initially, patients are anonymously registered at the current hospital where an EPR is created for each patient associated with a unique long-term anonymous EPR identifier (AEID) (Fig. 2).

For each subsequent hospitalization, each patient establishes a valid temporary anonymous identifier (TAPI) (Fig. 3) by which the patient can anonymously grant medical personnel EPR access according to the anonymity-preserving authentication and authorization (AAA) protocol (Fig. 4).

Intuitively, a naive way for P_i to anonymously convince S about ownership of a certain EPR is by using some kind of reference that anonymously links to the patient's EPRs. Let aid_i denote such reference. Assuming that such a reference aid_i is known only to the patient P_i and S, P_i could use it to authenticate himself or herself to S.

P_i could send aid_i encrypted to S along with a timestamp. S would then decrypt, check the timestamp, and authenticate the validity of aid_i by checking if there exists an EPR that is referenced by aid_i. The problem about such authentication is two-fold: 1) The EPR table referencing aid_i and the corresponding EPR must be kept secret. 2) The number aid_i is equivalent to a long-term secret user key, whereas secret keys should never be disclosed. An adversary getting hold of aid_i could obtain the same by encrypting aid_i along with a valid timestamp. Thus, this approach would not be appropriate. In the protocols shown in Fig. 2 and Fig. 3, anonymous authentication is achieved by exchanging blinded messages without submitting identifying information directly. Blinding refers to hiding data usually by multiplication of a secret secret number. This allows subsequent operations like exponential operations to be performed on the blinded data by another party while effectively hiding the original data.

In the rest of this paper, we assume that all cryptographic computations are in a finite field $\mathbb{Z}_p$ determined by a large prime p where α is a generator to p. Both p and α are public.

1)	$S \rightarrow P_i:$	$r_S = (\alpha \cdot N)^{x_S}, N$
2)	$P_i \rightarrow S:$	$E_S(r_S^{x_i}, N^{x_i}, N)$
	$S:$	$AEID_i = r_S^{x_i} \cdot (N^{x_i})^{-x_S} = \alpha^{x_S x_i}$

Figure 2: The $AEID_i$ initialization protocol for P_i

4.1 Initialization

The initialization protocol (Fig. 2) is utilized when a patient P_i for the first time is registered at the hospital and his or her EPR is created. In the protocol, $AEID_i$ is blindly established jointly by P_i and S, and being associated to the his or her new EPR.

In the first step, the patient randomly generates a private long-term key x_i, which should preferably be stored in tamper-proof hardware like a personal smartcard, and cannot be disclosed. Alternatively, it could be $x_i = h(P_i'\text{s password})$ where h denotes a secure hash function. The value α^{x_i} is essential for long-term anonymous identification of P_i, because the anonymous $AEID_i$ is computed by S according to $AEID_i = (\alpha^{x_i})^{x_S}$ where x_S is the private key of S. In this protocol, the disclosure of α^{x_i} is insignificant.

Initially, S generates a nonce N. Then S computes and submits $r_S = (\alpha \cdot N)^{x_S}$ and N to the patient P_i. Based on this, P_i computes $r_S^{x_i}$ and N^{x_i}, and returns $E_S(r_S^{x_i}, N^{x_i}, N)$. (The notation $E_X(m)$ denotes that a message m is encrypted with the public key of entity X.)

S receives the message from P_i, decrypts it and obtains $r_S^{x_i}$ and N^{x_i} and N. S verifies whether the nonce N matches with what was sent to P_i in the previous message. In case of no match, S aborts initialization. Otherwise, S computes

$$\begin{aligned} AEID_i &= r_S^{x_i} \cdot (N^{x_i})^{-x_S} = (\alpha \cdot N)^{x_S x_i} \cdot (N^{x_i})^{-x_S} \\ &= \alpha^{x_S x_i} \cdot N^{x_S x_i} \cdot N^{-x_i x_S} = \alpha^{x_S x_i} \end{aligned}$$

It is computationally infeasible to for S to obtain x_i due to the Discrete Logarithm Problem. Then S creates an empty EPR for P_i, and creates a new row in the AEID/EPR entry table linking the anonymous $AEID_i$ and the new EPR.

The difference of proposed initialization protocol from the Diffie-Hellman key agreement protocol [10] lyes in the fact that $AEID_i$ acts as an identifier, while the purpose of Diffie-Hellman is for two parties to establish secret keys that are not to be known by other than the two parties. Although that the table of $AEID_i$ should be kept secret at the EPR server, only knowledge of the private key x_i can provide EPR authorization, knowledge of the respective $AEID_i$ cannot.

4.2 TAPI establishment

When patients are getting hospitalized, a semi-trusted administrator (STA) provides related administrative tasks and coordination by assigning available and

1)	$S \rightarrow P_i:$	$r_S = \alpha^{x_S} \cdot N^{x_S}, N$
	$P_i:$	$TAPI_i = \alpha^{p_i}$
2)	$P_i \rightarrow S:$	$E_S(r_S^{x_i}, N^{x_i}, N, TAPI_i)$
	$S:$	$AEID_i = r_S^{x_i} \cdot (N_1^{x_i})^{-x_S}$
3)	$S \rightarrow P_i:$	$TAPI_i$

Figure 3: The TAPI establishment protocol

appropriate medical personnel to provide care to the patient. The patient may (or may not) identify themselves with names and SSN to the STA for administrational purposes like billing, but the STA will be unable to link any names to the records of the patients.

The protocol for establishing an temporary anonymous patient identifier (TAPI) (Fig. 3) is similar to the one described in the previous subsection, except that in this one, the patient P_i is anonymously authenticated towards $AEID_i$, and that the patient provides $TAPI_i$ by which the patient will be referenced throughout the current hospitalization.

Initially, S generates a nonce N, computes $r_S = (\alpha \cdot N)^{x_S}$, and sends r_S and N to P_i. P_i computes $r_S^{x_i}$ and N^{x_i}. P_i generates a large secret random number p_i, and computes $TAPI_i = \alpha^{p_i}$. Then P_i encrypts $r_S^{x_i}$, N^{x_i}, N, $TAPI_i$ with the public key of S.

The nonce N ensures S that message 2 is not a replay of a previous session, and blinds α^{x_S} (due to the Discrete Logarithm Problem). S decrypts the message, verifies correctness of the received nonce N, and computes

$$\begin{aligned} AEID_i &= r_S^{x_i} \cdot (N^{x_i})^{-x_S} = (\alpha \cdot N)^{x_S x_i} \cdot (N^{x_i})^{-x_S} \\ &= \alpha^{x_S x_i} \cdot N^{x_S x_i} \cdot N^{-x_i x_S} = \alpha^{x_S x_i} \end{aligned}$$

Then S checks in the AEID/EPR table whether there is an entry that matches $AEID_i$. If such entry exists, then S sends back $TAPI_i$ as an acknowledgment to STA and P_i to confirm that a record exists according to the request.

$TAPI_i$ is an anonymous identifier that identifies the patient P_i during the hospitalization where its validity is approved by S. However, in the proposed approach, it is not only for referencing the patient anonymously, but also to function as an anonymous ephemeral public key. During subsequent execution of the AAA protocol, the patient is authenticated by means of $TAPI_i$ (which is already approved by S), acting as a anonymous ephemeral public key where p_i is the corresponding private key. Correspondingly, both S and T are associated with respective public keys, although not anonymous.

For subsequent execution of the AAA protocol, the EPR server needs to make a temporary association between $TAPI_i$ and $AEID_i$ to facilitate subsequent quick database look-up. However, S must not disclose this association to ensure unlinkability.

1)	$P_i \rightarrow S:$	$N_{P_i}, TAPI_i$
2)	$S \rightarrow T:$	$N_S, [N_S, N_{P_i}, TAPI_i]_S$
3)	$T \rightarrow P_i:$	$N_T, [N_T, N_S, S]_T, \left[N_T, [N_S, N_{P_i}, TAPI_i]_S\right]_T$
4)	$P_i \rightarrow S:$	$[N_{P_i}, N_T, T]_{P_i}, \left[N_{P_i}, [N_T, N_S, S]_T\right]_{P_i}$
5)	$S \rightarrow T:$	$\left[N_S, [N_{P_i}, N_T, T]_{P_i}\right]_S$

Figure 4: The anonymity-preserving authentication and authorization protocol

4.3 Anonymity-preserving authentication and authorization

Fig. 4 shows the anonymous authentication protocol. The notation $[m]_X$ denotes entity X's signature of m. N_X denotes a nonce generated by X. The protocol is circular in the sense that the three parties, P_i, S, T, form a circle where all messages are sent in one direction. The messages flow according to $P_i \rightarrow S \rightarrow T \rightarrow P_i \rightarrow S$. Thus, each entity receives messages only from the preceding entity, and sends messages only to its succeeding entity.

In this protocol whenever a patient P_i needs to sign a message, he or she utilizes $TAPI_i = \alpha^{p_i}$ as a public key and p_i as a private key (see the previous subsection). Due to the form of the public/private key, the choice of possible signature schemes is restricted to an ElGamal-based signature scheme [8] or similar like Schnorr's signature scheme [9] or the Digital Signature Standard scheme [11] which are all based on the Discrete Logarithm Problem.

The protocol proceeds as follows:

1. P_i generates a nonce N_{P_i}, and sends $TAPI_i$, N_{P_i} to S.
2. S generates a nonce N_S, signs $[N_S, N_{P_i}, TAPI_i]$, and then sends $N_S, [N_S, N_{P_i}, TAPI_i]_S$ to T.
3. Upon receiving message 2, T extracts $[N_S, N_{P_i}, TAPI_i]$ and verifies the consistency of the message by comparing N_S of the signature with the unsigned N_S.
   If both match, T generates a nonce N_T, then signs $[N_T, N_S, S]$ and $\left[N_T, [N_S, N_{P_i}, TAPI_i]_S\right]$, and sends $N_T, [N_T, N_S, S]_T, \left[N_T, [N_S, N_{P_i}, TAPI_i]_S\right]_T$ to P_i.
4. Upon receiving message 3, P_i extracts $[N_T, N_S, S]$, and $\left[N_T, [N_S, N_{P_i}, TAPI_i]_S\right]$ by using public key of T. Then P_i extracts $[N_S, N_{P_i}, TAPI_i]$ using public key of S, and verifies the correctness of the challenge N_{P_i}. P_i then verifies that the message is consistent by checking that all three instances of N_T match, and that both instances of N_S match. If they are correct, P_i signs $[N_{P_i}, N_T, T]$ and $\left[N_{P_i}, [N_T, N_S, S]_T\right]$ with his private key p_i, and sends $[N_{P_i}, N_T, T]_{P_i}, \left[N_{P_i}, [N_T, N_S, S]_T\right]_{P_i}$ to S.
5. Upon receiving message 4, S extracts $[N_{P_i}, N_T, T]$ and $\left[N_{P_i}, [N_T, N_S, S]_T\right]$ using $TAPI_i$, the anonymous public key of P_i. S extracts moreover

$[N_T, N_S, S]$ by using the public key of T, and verifies the correctness of the challenge N_S. S then verifies that the message is consistent by checking whether both instances of N_{P_i} match with N_{P_i} of message 2, and whether both instances of N_T match.

If they are correct, S signs $\left[N_S, [N_{P_i}, N_T, T]_{P_i}\right]$, and sends $\left[N_S, [N_{P_i}, N_T, T]_{P_i}\right]_S$ to T.

6. Upon reception of message 5, T extracts $\left[N_S, [N_{P_i}, N_T, T]_{P_i}\right]$ by using public key of S and verifies that N_S match the values of N_S of message 2. Then T moreover extracts $[N_{P_i}, N_T, P]$ from $[N_{P_i}, N_T, T]_{P_i}$ by using $TAPI_i$, and verifies that N_{P_i} match N_{P_i} of message 2, and the correctness of the challenge N_T.

The protocol complies to that the participating entities authenticates each other in such a way the P_i remains anonymous. The team T is represented as an entity, but correspondingly consists of a number of collaborating team members. In order for S and P to be certain that indeed a certain minimum number of the participants of T actually are present, and that not just one individual is acting on behalf of T, a threshold group-oriented cryptosystem should be employed by T. In a threshold group-oriented cryptosystem, a group is represented by a public key, but a specific minimum number of the participants must collaborate to decrypt messages encrypted by the threshold-based public key [6, 7].

5 Security analysis

5.1 The TAPI protocol

First, we need to show that using $TAPI_i$ cannot be associated with P_i and therefore utilization of $TAPI_i$ as the patient's public key will preserve patient's anonymity. According to the algorithm, the patient generates the private p_i randomly. Therefore, the corresponding $TAPI_i = \alpha^{p_i}$ is also random. The association between $TAPI_i$ and patient P_i (via $AEID_i$) is sent encrypted to S with the public key of S. Assuming that the employed public key cryptosystem is secure, we can conclude that $TAPI_i$ preserves the patient anonymity, and unlinkability as it is defined in the beginning of this paper is provided.

In both the AEID and TAPI protocols, the value of $AEID_i$ is blindly established where P_i signs by means of his or her private key x_i the blinded value r_S issued by S. Since only S holds the secret key x_S, S can recover $AEID_i$ as an application of the ElGamal cryptosystem [8].

5.2 The AAA protocol

The function of the AAA protocol is for the three parties, P_i, S, and T, to mutually authenticate each other. P_i is anonymously represented by his or her public key $TAPI_i$. Each party generates a nonce acting as a challenge. For all three

parties, the two other parties sign the nonce issued by a first party. Subsequent verification of the signatures establishes the authenticity of the participants. Each signature includes the nonce issued by the originating party in order to prevent that an adversary can obtain illegitimate signatures [12, p. 109].

In a given session, each participant issues a unique random nonce in messages 1–3. Each of the succeeding messages is cryptographically linked to each other by means of the nonces. Redundancy of nonces in each messages (except for message 1) ensures that each message is consistent. P_i receiving message 3 can certify the authenticity of S and T according to the signed challenge N_{P_i} originated at P_i. Moreover, the nonces N_S and N_T ensures that the consistency of message 3 can be certified. By *consistency*, we mean that the parts of the message must correspond, where certification of consistency reveals that a message has been tampered with by modification of substitution of a part of the message. Likewise S can, upon receiving messages 1 and 4, certify P_i and T and certify the consistency of the message. Finally, T upon receiving messages 2 and 5, certifies P_i and S.

6 Conclusion

In this paper, we have addressed the need of keeping patient identities anonymous in regard to their medical data, in contrast to only keeping their medical data confidential. This is essential in cases where patients wish that their identities (like names, addresses, personal identity numbers, etc.) cannot to be associated with information about for example related diseases, physical or genetic defects, drug addictions, etc., stated in their respective patient records. To achieve that personal identity information is not to be linked with disclosed medical data, we have proposed a scheme that enables patients themselves to *anonymously* grant medical teams authorization to access their EPRs without revealing their identities to the teams, by instead using certifiable anonymous identifiers. Another benefit of this scheme is that it allows patients to exert control over their own medical records.

References

[1] T. Rindfleich. Privacy, information technology and health care. Communications of the ACM, 1997, Vol. 40, No. 8.

[2] A. Tveit et al. Anonymization of General Practioner's Patient Records. Proc. of HelseIT'04, 2004.

[3] L. Sweeney. Guaranteeing anonymity when sharing medical data, the datafly system. Journal of the American Medical Informatics Association, 1997.

[4] A. Kalam et al. Smartcard-based Anonymization. In proc. CARDIS, 2004, pp. 49-66.

[5] A. Kalam et al. A generic approach for healthcare data anonymization. Proc. of the 2004 ACM workshop on Privacy in the electronic society, 2004, pp. 31–32, ACM Press.

[6] Y. Desmedt. Threshold cryptosystems. Advances in Cryptology - Auscrypt '92, Springer-Verlag, 1993, LNCS 718, pp. 3 – 14.

[7] S. Saeednia, H. Ghodosi. A self-oriented group-oriented cryptosystem without a combiner. Proc. of the 4th Australasian Conference on Information Security and Privacy, 1999, Springer-Verlag, pp. 192–201.

[8] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, 1985, pp. 469-472.

[9] C. P. Schnorr. Efficient identification and signatures for smart cards. CRYPTO '89: Proceedings on Advances in cryptology, 1989, pp. 239–252, Springer-Verlag New York, Inc.

[10] W. Diffie, M. E. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, Vol. IT-22, No. 6, pp. 644–654, 1976.

[11] National Institute of Standards and Technology. FIPS PUB 186-2: Digital Signature Standard (DSS). NIST, 2000.

[12] C. Boyd, A. Mathuria. Protocols for authentication and key establishment. Springer-Verlag, 2003.

Paper G

EPR Access Authorization of Medical Teams
Based on Patient Consent

Sigurd Eskeland and Vladimir Oleshchuk

EPR Access Authorization of Medical Teams Based on Patient Consent

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract
Electronic patient records (EPR) may contain highly confidential and personal medical information. It is therefore essential that medical data is properly protected and managed. Today, it is widely recognized that patients have a right to self-determination and to exert control of their own medical data by consent. In this paper, we present a cryptographic EPR access authorization scheme that incorporates patient consent as a basis for granting EPR access to medical teams or practitioners. This ensures that only the medical practitioners specified by a consenting patient are granted EPR access. If a patient is unconscious, the variation of the scheme allows an emergency or security team to act on behalf of the patient.

1 Introduction

With the emergence of information technology in health care, there has been extensive focus on security and confidentiality issues of electronic patient records (EPR) in medical environments [1–4]. An important issue here concerns proper access control. A basic criterion for this is legitimacy, i.e., only medical personnel providing medical care to a given patient (or patients) should access only the necessary medical data of the concerning patient they are providing care to [6]. Another significant security issue concerns secure and confidential management, handling and storage of personal medical information [1].

In a typical medical information scenario, electronic patient records could be stored in EPR servers that are managed and controlled by one or few security administrators. These administrators would normally possess all privileges with respect to the patient data. They would perform functions such as authorizing and assigning medical practitioners access to the EPRs of the concerning patients that are to be provided care for. Consequently, each security administrator would have full access to all personal medical data. However, as patient records may contain highly sensitive and confidential personal information, it is very important to ensure that such information remains confidential. In this scenario, the patients have no actual control over their medical data and are in practice left no other option than to simply

trust that their data will not be disclosed to illegitimate personnel nor manipulated. However, patient consent identifies what nowadays has generally been recognized as patients' rights to exert control over their own medical data [5–8]. Patient consent has today become an important principle in medical ethics and access control policies. Even though this has lately been a widely recognized aspect concerning patients' self-determination and right to exert control over their own personal medical data, patient consent is in practice enforced by means of filling out paper forms. Since this does not impose an actual obstacle against illegitimate EPR access, it is therefore important that patient consent should be integrated in medical access control systems.

As medical data in general should be protected from disclosure to unauthorized personnel, certain data are more sensitive than others. Since medical records may possibly contain information about AIDS/HIV status, sexual transmittable diseases, emotional problems, psychiatric illnesses, sexual divergencies, genetic predispositions to diseases, information about toxic addictions, and so on [1], it essential that such information should be protected from disclosure including to security personnel except when legitimately needed by medical practitioners. To ensure the privacy of medical data, the EPRs could be stored encrypted at the EPR server. Alternatively, assuming that the EPR is arranged into blocks or modules, a proper arrangement could be that only certain EPR modules containing particulary sensitive data are encrypted. Encryption imposes, however, the problem of secure key storage and management. For example, if a cryptokey is revealed, the encrypted data can be decrypted and revealed. If a cryptokey is lost, the data is lost. A straight-forward solution is that one or few security administrators would control all EPR cryptokeys. Due to the fact that security administrators would be individually entrusted with the responsibility of managing possibly thousands of secret EPR keys, which could impose a considerable risk of human error, fraud, attacks and possibly high workloads.

A naive and insufficient solution could be to use a threshold-based (t, n) secret sharing scheme where $t \leq n$ and the corresponding key is split into n secret shares [10]. The shares are distributed to n authorities, so that each individual holds one share. The secret key can only be reconstructed when at least t of the participants pool their shares together. However, there are at least three shortcomings with this approach: 1) The same secret key is associated to all EPRs. 2) When reconstructed, the single secret key is revealed once and for all. 3) The participants must reveal their secret shares to each other in order to reconstruct the secret key. Thus, there is no confidentiality regarding the secret shares.

In this paper, we present a cryptographic access authorization scheme that incorporates the concept of patient consent. We consider the function of patient consent to be equivalent with the function of granting or authorizing EPR access.

Note that EPR access could be granted to individuals, except to the patients themselves, instead of teams like to a specific doctor (specialist, general prac-

titioner, etc.). This provides proper distribution of trust since the patient is in charge of disclosing his or her EPR by consent. However, the medical data cannot be accessed by the patients themselves without special arrangements.

We assume that each EPR is encrypted by a unique and distinct key unknown to all participants including the pertaining patient. The scheme provides secure and confidential establishment of EPR cryptokeys for subsequent decryption of the pertaining medical records. There are no cryptokey tables, but the cryptokey for a given EPR is temporarily restored at the EPR server for each session by means of the consenting patient holding a secret user key (not the EPR cryptokey) in conjunction with the EPR server. The scheme is secure and prohibits deduction of secret user keys or EPR cryptokeys. Accordingly, medical data is protected due to that electronic patient records (or modules) can be stored encrypted at an EPR server, prohibiting medical data to be disclosed without the collaboration of the consenting patient and a medical team. *Encryption* of updated medical data could be done at the EPR server or by medical practitioners by means of a corresponding public key.

The EPR cryptosystem presented in [9] seems to be the only EPR cryptosystem incorporating patient consent for EPR authorization of medical teams. However, a serious security weakness about this cryptosystem is that the EPR server does not have an active function in EPR cryptokey reconstruction, enabling a colluding patient and team to reveal the secret EPR cryptokey independently of the EPR server.

The rest of the paper is organized as follows: In Section 2, we give a brief introduction to threshold cryptography. In Section 3, we present the cryptographic scheme. In case a patient is unconscious, he or she would not be in a position to actively and consciously grant anybody EPR access. In Section 4, we present a variant of this scheme for the emergency case, allowing a coalition of security administrators or emergency team to grant medical personnel EPR access on behalf of the patient.

2 Group-orientation and threshold cryptosystems

The motivation of threshold cryptosystems is to enforce that a given minimum number of participants from a group are required to compute a cryptographic operation, in contrast of requiring *all* the participants, i.e., a fixed set of users, for this. Thus, the term *threshold* denotes the minimum number of participants of required for this. This is desirable in scenarios where group consensus is required, for example that the holder or originator of some sensitive information like a secret key, is only willing to disclose it as result of the agreement and consensus of a given number of associated participants. Accordingly, it is prevented that single individuals can obtain the secret on their own. As a practical example, we can consider access to a bank vault where it is not desirable that one person alone would possess and control the key to the vault due to the risk of fraud, robbery and extortion, but the

participation of at least 2 or 3 persons out of for instance 4, each holding a unique and secret key, should be required in order to unlock the vault. This is a desirable requirement in security systems involving consensus and collaboration of several participants.

Typical threshold-oriented applications are threshold decryption and threshold signatures. A threshold decryption cryptosystem is a cryptosystem requiring an arbitrary composed subset of a minimum number of participants of a given group to collaboratively perform decryption. Represented by a public key, outsiders can confidentially address the group. Only by collaboration where the active group members are providing partial computations, the encrypted message can be decrypted [12–14]. Likewise, regarding threshold signatures [15, 16], only a minimum subset of the team can compute signatures due to the threshold requirement.

3 EPR access authorization based on patient consent

In this section, we present the cryptographic EPR access authorization scheme. It assumes that the medical records are stored encrypted on a server. Each EPR is encrypted by a unique secret key and there are *no* cryptokey tables. The proposed scheme has mainly two purposes: The first is to enable patients to securely grant EPR access to medical teams and medical practitioners. The second purpose is to provide secure and temporarily reconstruction of the secret cryptokey for a given EPR at the EPR server from the process of a patient granting a medical team access to his or her EPR.

The scheme enables reconstruction of a predefined EPR cryptokey (which thus is the same for each session), based on the computations involving the secret keys of the pertaining patient and the EPR server. The server subsequently decrypts the given EPR. The protocol prevents disclosure and deduction of restored EPR cryptokeys to any party other than the EPR server. It moreover prevents that any secret inputs or keys of the participants can be deduced by any participating or external party.

The patient grants a medical team EPR access by basically generating a secret cryptographic challenge in agreement with the public key of the pertaining team. The EPR server will only be able to reconstruct the secret EPR cryptokey provided a valid response. Since only associated members of the addressed medical team can collaboratively provide the correct response to the challenge, this ensures that no one other than the genuine team can obtain access to the patient's EPR. Otherwise, the pertaining EPR cryptokey cannot be restored. An eligible minimum number of active team participants is defined by applying a threshold mechanism.

3.1 Protocol initializations

A trusted authority (TA) is responsible for providing the required public key infrastructure. Let $\mathcal{U} = \{P_1, \ldots, P_n\}$ denote a medical team of n members. The TA defines the minimum number of active participants t that are required in order to obtain the EPR access granted by the patient. This subcoalition is denoted $T \subseteq \mathcal{U}$ where $|T| \geq t$. According to the Shamir secret sharing scheme [10], the TA generates a unique secret polynomial of degree $(t-1)$:

$$f(x) = \sum_{j=0}^{t-1} a_j \, x^j$$

that represents the team $\mathcal{U}$. The TA computes one personal long-term secret share for each team member as follows: For each $P_i \in \mathcal{U}$, the TA arbitrarily selects a input x_i from $\mathbb{Z}_q$, and computes the secret user share

$$s_i = f(x_i) \pmod{q}$$

where q is a large public prime. The team $\mathcal{U}$ is externally represented by the public key

$$y = \alpha^{a_0} \pmod{p}$$

where $a_0 = f(0)$ and α is a generator to $\mathbb{Z}_p$. Note that $p = 2 \cdot q + 1$ is a large public prime.

Let S denote the EPR security server and G_i denote the patient (the granting entity). The TA moreover provides the EPR server S with the secret key k_s and each patient G_i with the secret key k_i where $k_s, k_i \in \mathbb{Z}_q$. The TA computes the secret EPR cryptokey

$$K_i = \alpha^{k_s \cdot k_i} \pmod{p}$$

by which the TA encrypts the EPR of G_i by means of a proper cryptographic algorithm. The TA deletes K_i subsequently.

3.2 Protocol description

In this section, we describe the cryptographic EPR access authorization scheme. This is moreover presented in Figure 1, and goes as follows:

Step 1. The protocol is initiated by S that generates the secret random numbers $r_1, r_2 \in \mathbb{Z}_q$, and computes for G_i

$$a_s = \alpha^{r_1} \pmod{p}, \quad b_s = \alpha^{k_s} \alpha^{r_1 \cdot r_2} \pmod{p}$$

Step 2. The patient G_i grants EPR access to a medical team $\mathcal{U}$ (the grantee) by means of the team's public key y. G_i generates a random secret number $r_i \in \mathbb{Z}_q$, computes and returns (c_i, d_i, R_i, y) to S where

$$c_i = b_s^{k_i} \pmod{p}, \quad d_i = y^{r_i} \, a_s^{k_i} \pmod{p} \quad \text{and} \quad R_i = \alpha^{r_i} \pmod{p}$$

The public key y is included in the message, indicating to the medical team that G_i claims to be the grantee. Whether y is the *genuine* key applied in the computation of d_i, is certified according to the correctness of K_i computed in Step 5. Note that since y^{r_i} is an unknown factor of d_i and r_i is secret, this can only be resolved by the partial computations of team members holding the secret user shares, collectively computing $(\alpha^r)^{a_0}$ which corresponds to y^{r_i}.

Step 3. S checks that y is a public key of a genuine and approved medical team or medical practitioner. Otherwise, G_i could grant EPR access to illegitimate persons, knowingly or unknowingly. If y is accepted, S broadcasts the challenge

$$u_s = (\alpha \cdot R_i)^{r_2} \pmod{p}$$

Otherwise, terminate.

Step 4. To correctly respond to the challenge u_s, the partial computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required. Each team member $P_j \in T$ receives u_s, and computes and returns the partial computations $z_j = u_s^{s_j} \pmod{p}$.

Step 5. Key computation. S applies Lagrange interpolation to the partial computations of T according to

$$Y_i = \prod_{j \in I_T} z_j^{b_j} \pmod{p} \quad \text{where} \quad b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x_j}{x_j - x_i} \pmod{q}$$

and $I_T = \{i \,|\, P_i \in T\}$. By means of the secret r_2, only S is capable of reconstructing the EPR cryptokey as follows:

$$\begin{aligned} K_i &= \alpha^{k_i \cdot k_s} = c_i \cdot Y_i \cdot (d_i \cdot y)^{-r_2} \pmod{p} \\ &= (\alpha^{k_s \cdot k_i} \cdot \alpha^{r_1 \cdot r_2 \cdot k_i}) \cdot (\alpha^{r_i \cdot r_2 \cdot a_0} \cdot \alpha^{r_2 \cdot a_0}) \cdot (y^{r_i} \cdot \alpha^{r_1 \cdot k_i} \cdot y)^{-r_2} \\ &= (\alpha^{k_s \cdot k_i} \cdot \alpha^{r_1 \cdot r_2 \cdot k_i}) \cdot (y^{r_i \cdot r_2} \cdot y^{r_2}) \cdot (y^{-r_i \cdot r_2} \cdot \alpha^{-r_1 \cdot r_2 \cdot k_i} \cdot y^{-r_2}) \end{aligned}$$

by which it subsequently decrypts the pertaining EPR. The EPR can be securely transferred to $T \subseteq \mathcal{U}$, for example by encrypting it with public key of $\mathcal{U}$.

Given that r_2 is secret prohibits anyone but S to obtain K_i. Note that in Step 2, G_i indicates the public key of the grantee. Since this is included in the key reconstruction phase, the correctness of K_i is ensured according to that only $T \subseteq \mathcal{U}$ represented by y can provide the correct response in Step 4.

Note that the function of this protocol is very different from the function of key establishment protocols which provide secure establishment of non-predefined random secret shared keys over insecure networks. Participants of such protocols are usually authenticated towards their public keys. Our protocol differs since its function is to securely establish a *predefined* secret key whereof its correctness implies authenticity of the participants. For example,

only G_i, holding k_i, can contribute to establish the correct K_i, disregarding the emergency case. Further user/key authentication is thus not required.

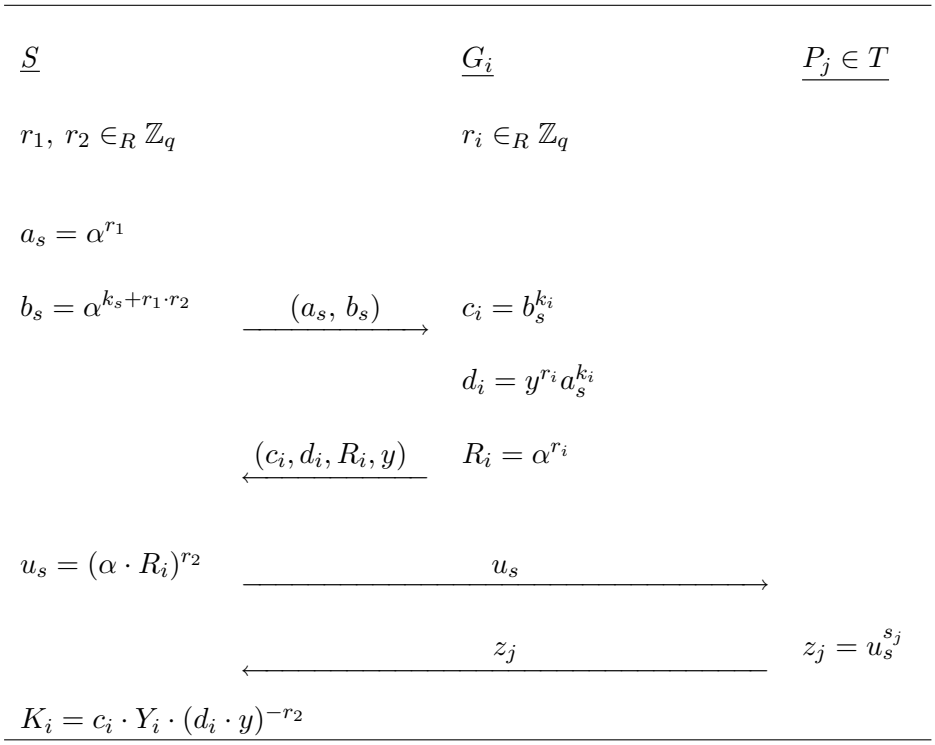

Figure 1: EPR cryptokey reconstruction due to the proposed scheme. All computations are in $\mathbb{Z}_p$.

3.3 Security discussion

In this subsection, we will show that the proposed protocol preserves both authenticity and key secrecy.

Authenticity. The protocol provides a legitimate user to securely reconstruct the secret EPR cryptokey associated to him or her at the EPR server and no one else. Thus, if an illegitimate user tries to establish a given key, it will fail. It is essential that the protocol preserves the authenticity of the users, resisting any masquerading attack so that no entity (internal or external) may successfully masquerade as another entity. Since the goal is to establish a fixed secret EPR cryptokey, explicit user authentication is not required since the key is established as a function of the secret user keys held by the participants and is therefore implicitly provided.

Like most cryptographic authentication protocols, user authentication is provided on the assumption that only the genuine user and no one else is holding a specific secret whereof the genuineness of his or her identity is based. The protocol provides the ability for the user to prove that he or she actually holds

the specific secret according to the correctness of the result. Accordingly, the protocol must prevent that anybody else can establish or obtain the correct result and therefore illegitimately obtain access to a patient's EPR.

Note that (a_s, b_s) are cryptographically bound to the secrets r_1 and r_2 only known by S. This binding prevents replay attacks where an adversary attempts to successfully run the protocol by masquerading. An adversary replaying the numbers c_i, d_i, R_i from a former session would cause inconsistency in the key recovery phase since r_2 is distinct and unique for each session, and only the genuine value of r_2 can resolve the EPR cryptokey.

Key secrecy. There are two aspects regarding key-secrecy. First, it is required that no secret user keys or secret user shares can be deduced from the messages. Secondly, it must be infeasible to deduce EPR cryptokeys for anybody except S.

Regarding the first key secrecy requirement, no user input must be revealed from the computations. This is obtained due to the Discrete Logarithm Problem that protects the secret key k_s of S in Step 1, the secret key k_i of G_i in Step 2 and the secret user shares of each $P_j \in T$ in Step 4.

Considering the secrecy of the EPR cryptokey, disclosure of α^{k_s} must be prevented, otherwise a patient could compute $K_i = (\alpha^{k_s})^{k_i}$. Regarding $a_s = \alpha^{k_s}\alpha^{r_1 \cdot r_2}$ and $b_s = \alpha^{r_1}$ in Step 1, due to the Diffie-Hellman assumption, it is computationally infeasible to obtain $\alpha^{r_1 \cdot r_2}$ given α^{r_1} and α^{r_2} where r_1 and r_2 are unknown. However, an adversary could try to attack the protocol by returning α^{-r_1} to S in Step 3. Since S would then compute $u_s = \alpha^{-r_1 \cdot r_2}\alpha^{r_2}$ in Step 4, the adversary would only obtain $a_s \cdot u_s = \alpha^{k_i} \cdot \alpha^{r_2}$ where α^{r_2} is unknown. Thus, the attack would fail.

The EPR cryptokey is the first factor in $c_i = \alpha^{k_s \cdot k_i} \cdot \alpha^{r_1 \cdot r_2 \cdot k_i}$. Likewise, it is protected by the unknown second factor $\alpha^{r_1 \cdot r_2 \cdot k_i}$ where it is computationally infeasible to obtain $\alpha^{r_1 \cdot r_2}$.

Note that α^{k_i} could be a public key of G_i though it would have no function in this protocol. An adversary would have no use of this due to that knowledge of the secret k_i is required for the exponentiations for computing c_i and d_i in Step 2.

4 The emergency case

There could be situations when patients are in a coma, or situations of car accidents, fire, terrorist acts, etc., where patients may be unconscious and may therefore not be able to actively grant any medical practitioners access to his or her EPR. In this section we describe a modified version of the protocol presented in the previous section to handle such emergency cases. In emergency cases, a coalition of security administrators or an emergency team could act on behalf of the patient to grant EPR access. Note that there should be a minimum threshold in order to prohibit that any single individual may solely grant or obtain access to personal medical data.

In normal situations, the patient would by means of his or her secret key grant any team access to his or her EPR. For the emergency case, each patient could be represented by an associated public parameter or identifier that the security team would use to reference the patient.

The emergency case could be handled as follows: The TA defines the minimum threshold t' of security administrators that is required to actively grant on behalf of a pertaining patient that is disabled. The TA generates a random secret polynomial $g(x)$ of order $(t'-1)$ that represents the team of security administrators SA. The TA computes for each administrator $A_i \in SA$ a secret share according to

$$t_i = g(i) \pmod{q}$$

The secret keys of each patient G_i is computed according to

$$k_i = g(h(G_i)) \pmod{q}$$

where G_i denotes the identity of the patient and h is a secure one-way function.

The protocol is as the previous except that SA coalition acts on behalf of the patient which introduces a second team aspect. The protocol goes as follows:

Step 1. The protocol is initiated by S that generates the secret random numbers $r_1, r_2 \in \mathbb{Z}_q$, and forwards to SA the challenges

$$a_s = \alpha^{r_1} \pmod{p}, \quad b_s = \alpha^{k_s + r_1 \cdot r_2} \pmod{p}$$

Step 2. The SA grants a medical team $\mathcal{U}$ EPR access on behalf on G_i by means of the team's public key y. Each $A_j \in SA$ generates a random secret number $r_j \in \mathbb{Z}_q$, and computes and returns (c_j, d_j, R_j) to S where

$$c_j = b_s^{t_j \cdot b_j} \pmod{p}, \quad d_j = a_s^{t_j \cdot b_j} y^{r_j} \pmod{p} \quad \text{and} \quad R_j = \alpha^{-r_j} \pmod{p}$$

Note that

$$b_j = \prod_{\substack{k \in I_{SA} \\ j \neq k}} \frac{h(G_i) - k}{j - k} \pmod{q}$$

and $I_{SA} = \{j \mid A_j \in SA\}$. Also note that the computations of SA agree to Lagrange interpolation on exponents (Step 3) which corresponds to applying k_i as an exponent to (a_s, b_s) as in Section 3.

Step 3. S receives the messages from SA and computes Lagrange interpolation on the exponents by multiplication

$$c_i = \prod_{j \in I_{SA}} c_j \pmod{p}, \quad d_i = \prod_{j \in I_{SA}} d_j \pmod{p} \quad \text{and} \quad R = \prod_{j \in I_{SA}} R_j \pmod{p}$$

and forwards the challenge R to $\mathcal{U}$.

Step 4. To correctly respond to the challenge R, the partial computations of a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required. Each $P_j \in T$ receives R, and computes and returns $Y_j = R^{s_j} \pmod p$.

Step 5. Due to the fact that r_i is secret, S is required to obtain Y_i by Lagrange interpolation of the partial computations of T according to

$$Y_i = y^{r_i} = \prod_{j \in I_T} Y_j^{b_j} \pmod p \quad \text{and} \quad b_i = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x_j}{x_j - x_i} \pmod q$$

where $I_T = \{i \mid P_i \in T\}$. Finally, S reconstructs the secret key

$$K_i = c \cdot (d \cdot Y_i)^{-r_2} = (\alpha^{k_s \cdot k_i} \cdot \alpha^{r_1 \cdot r_2 \cdot k_i}) \cdot (y^{-r_i \cdot r_2} \alpha^{-r_1 \cdot k_1 \cdot r_2}) \cdot (y^{r_i \cdot r_2}) = \alpha^{k_i \cdot k_s} \pmod p$$

by which it subsequently decrypts the pertaining EPR.

5 Conclusion

In this paper, we have presented a cryptographic EPR access authorization scheme that incorporates patient consent as a basis for granting EPR access. This ensures that only the medical practitioners specified by a consenting patient are granted EPR access. If a patient is unconscious, a variation of the scheme allows an emergency or security team to act on behalf of the patient.

The security scheme assumes that electronic patient records (or specific parts of patient records) are stored encrypted at the EPR server and each EPR is encrypted with a unique and secret key. The key management problem is precluded due to the fact that there are no cryptokey tables and no one, including patients, hold or can obtain the cryptokey that can decrypt his or her EPR. However, each patient holds a long-term secret user key. Instead, the protocol enables secure reconstruction of a secret EPR cryptokey at the EPR server from the cryptographic interaction between the EPR server and the pertaining patient granting a medical team access to his or her EPR. This allows the EPR server to subsequently decrypt the pertaining EPR. The scheme is secure in the sense that it prohibits that secret user key and EPR cryptokeys can be deduced and disclosed.

References

[1] T. Rindfleich. Privacy, information technology and health care. Communications of the ACM, 1997, Vol. 40, No. 8.

[2] J. Biskup, G. Bleumer. Cryptographic protection of health information: cost and benefit. International Journal of Bio-Medical Computing, Vol. 43, pp. 61-67, 1996.

[3] G. Serour. Confidentiality, privacy and security of patients' health care information: FIGO Committee for the Ethical Aspects of Human Reproduction and Women's Health. International Journal of Gynecology & Obstetrics, Vol. 93, Iss. 2, pp. 189-190, 2006.

[4] Committee on Maintaining Privacy and Security in Health Care Applications of the National Information Infrastructure. For the Record: Protecting Electronic Health Information. National Academies Press, ISBN-10: 0-309-05697-7, 1997.

[5] P. A. B. Galpottage, A.C. Norris. Patient consent principles and guidelines for e-consent: a New Zealand perspective. Health Informatics Journal. SAGE Publications, Vol. 11, No. 1, pp. 5-18, 2005.

[6] American Medical Association. Patient Confidentiality. See http://www.ama-assn.org/ama/pub/category/4610.html.

[7] E. Coiera, R. Clarke. e-Consent: The design and implementation of consumer consent mechanisms in an electronic environment. Journal of the American Informatics Association, Vol. 11, pp. 129-140, 2004.

[8] J. Bergmann et al. An e-consent-based shared EHR system architecture for integrated healthcare networks. International Journal of Medical Informatics, Vol. 76, No. 2-3, pp. 130-136, 2007.

[9] S. Eskeland. Access control by secure multi-party EPR decryption in the medical scenario. IASTED Int. Conf. on Communication, Network, and Information Security '06, pp. 99-103, ACTA Press, 2006.

[10] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612-613, 1979.

[11] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469–472, 1985.

[12] Y. Desmedt, Y. Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307 – 315, Springer-Verlag, 1990.

[13] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, vol. 547, pp. 522 – 526, Spring-Verlag, 1991.

[14] S. Saeednia, H. Ghodosi. A self-oriented group-oriented cryptosystem without a combiner. Proc. of the 4th Australasian Conference on Information Security and Privacy, Springer-Verlag, pp. 192–201, 1999.

[15] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings, Vol. 141, No. 5, pp. 307-313, 1994.

[16] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders. Eurocrypt '94, pp. 194-204.

Paper H

Secure Team-Based EPR Access Acquisition in Wireless Networks

Sigurd Eskeland and Vladimir Oleshchuk

Secure Team-Based EPR Access Acquisition in Wireless Networks

Sigurd Eskeland Vladimir Oleshchuk
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
{sigurd.eskeland, vladimir.oleshchuk}@uia.no

Abstract
Electronic patient records (EPR) may contain highly confidential and sensitive medical data, and it is therefore essential that such information is properly protected. Medical teams that are providing care to a patient have a legitimate need to access the medical data of the concerning patient, and is a valid criterion for medical professionals to obtain access to such data. Group consensus could qualify as a basis for trust, and hence act as a criterion for a medical team to acquire access to the required medical data.
In this paper, we present three closely related threshold-oriented cryptographic protocols providing secure team-based EPR access acquisition, based on the consensus of a minimum number of associated medical practitioners. The schemes are broadcast-oriented, and are thus well-suitable for wireless networks. All schemes do also provide secure transfer of medical data.

1 Introduction

Today, personal medical data is commonly incorporated on an electronic platform so that the electronic patient record (EPR) is replacing the old paper-based patient record. Due to the sensitive nature of medical data together with the increased availability of such data on networked computer systems, the confidentiality and security aspects of medical information in EPR systems has become an essential issue [1–5].
A major issue concerns enforcement of proper access control. The sensitivity of personal health information necessitates that such information remains undisclosed except concerning legitimate purposes. Thus, a fundamental criterion for provision of access to medical data is the actual legitimacy of whom access is to be granted. Reasonably, this could be in agreement with the need-to-know-principle, i.e., that only medical personnel providing medical care to a given patient (or patients) should only be able to access the medical data of the concerning patients they are providing care to [1]. Thus, EPR access should be confined to that which is necessary to provide proper medical care. Access to medical data could therefore be denied to anybody, including medical practitioners, that does not have a legitimate need to access such data.

An access control policy based on legitimacy could for example be maintained by: 1) Patient consent. The patient must consent to that one or more medical practitioners may access his or her EPR [6, 7]. 2) One or more security administrators are required to grant, on behalf of a patient, one or more medical practitioners access to the EPR of this patient. A combination of these could be relevant considering emergency cases when a patient could be unconscious and is unable to authorize EPR access.
In the medical scenario, medical care is typically provided by medical personnel organized in teams. Consensus among a given number of associated medical practitioners could qualify as a reasonable criterion for authorization of EPR access. A medical team, consisting of a number of individuals working for the same cause, could thus be recognized as a proper basis for trust. Such a criterion would accordingly prevent that single individuals could illegitimately obtain access to EPRs unless holding special privileges.
In this paper, we present three closely related threshold-oriented cryptographic schemes that enable medical teams to acquire access to EPRs as a function of group consensus of an arbitrary predefined minimum number of participants. The protocols are broadcast-oriented, and are therefore well-suitable for wireless networks. Protocol 1 deals with the case where data is stored as plaintext on the EPR server, but provides secure EPR acquisition and transfer to a corresponding team.
It could be desired to preserve a more overall level of information confidentiality, not only with respect to access control (i.e., to whom EPR access should be granted), but also concerning the long-term storage of the medical data. For example, it could be argued that long-term encryption of medical data may provide an increased level of confidentiality than if stored as plaintext. Nevertheless, this assumes that the corresponding cryptokeys would have to be "out of reach" since encryption imposes the problem of secure key management and key storage. If a cryptokey is compromised, its encrypted data would correspondingly be considered compromised. Storage and management of cryptokeys would therefore impose a potential security risk since that whoever that is controlling the cryptokeys also controls the corresponding data.
In this context, we propose two related schemes (Protocol 2 and 3) where it is assumed that the EPRs are stored as cryptotext at the EPR server. We assume that each EPR is encrypted by a unique long-term secret EPR cryptokey that is unknown to all group participants (i.e., the members of the medical teams) including the pertaining patients. Protocol 2 and 3 provide privacy-preserving EPR cryptokey reestablishment for subsequent decryption and restoration of the of the pertaining EPR. This is achieved by the computations and collaboration of a predefined minimum (i.e., the threshold) of the participants of the concerning medical team and without revealing the long-term EPR cryptokey. There are no cryptokey tables that has to be securely maintained. The schemes prohibit deduction and disclosure of the secret EPR cryptokeys and the threshold

requirement provides the essential distribution of trust for acquiring EPR access.

2 Threshold cryptosystems

The motivation of threshold cryptosystems is to enforce that a given minimum number of participants from a group are required to compute a cryptographic operation, in contrast of requiring *all* the participants, i.e., a fixed set of users, for this. The term *threshold* denotes the minimum number of participants of the group or team required to carry out the threshold computation. This is desirable in scenarios where group consensus is required, for example that the holder or originator of some sensitive information like a secret key, is only willing to disclose it as result of consensus of a given number of associated participants. Accordingly, it is prevented that single individuals can obtain the secret on their own.
As a practical example, we can consider access to a bank vault. It may not be desirable that individuals could solely access the vault due to the risk of fraud, robbery and extortion. The safety would be considerably improved by a threshold-oriented security system, where the participation of at least 2 or 3 persons out of for instance 4, each holding a unique and secret key, should be required in order to unlock the vault. In threshold cryptosystems, each active participant carries out some partial computation that they succeedingly "pool" together in order to compute the threshold computation. Enforcement of the threshold requirement is desirable in security systems involving consensus of several participants.
Threshold cryptosystems are commonly based on the Shamir secret sharing scheme [14]. Typical threshold-oriented applications are threshold decryption and threshold signatures. A threshold decryption scheme is a cryptosystem requiring the partial computations of an arbitrary composed subset of t of n associated participants to carry out decryption. Represented by a public key, outsiders can confidentially address the group. Only by collaboration where the active group members are providing the partial computations, the encrypted message can be decrypted [9–11]. Likewise, regarding threshold signatures [12, 13], only a minimum subset of the team can compute signatures due to the threshold requirement. A broadcast-oriented conference key establishment scheme based on a threshold secret sharing was proposed in [15].

3 Secure acquisition of plaintext medical data

In this section, we propose a threshold-oriented EPR access acquisition scheme, where the medical data is stored as plaintext on the EPR server. The scheme is basically a broadcast-oriented threshold decryption scheme based on the ElGamal public key cryptosystem [16], and is thus suitable for wireless networks. By means of threshold decryption, data can be

securely transferred from one party to a team of associated users without prior establishment of or knowledge of a secret key. It requires that at least t associated participants or team members collectively compute and share their partial decryptions in order to decrypt pertaining encrypted messages. Accordingly, it precludes that single individuals can obtain the secret data on their own.

3.1 Protocol 1

Initialization. A trusted authority (TA) is required to set up the scheme, i.e., to provide secret user keys and a public key representing each respective team. Let $\mathcal{U} = \{P_1, P_2, \ldots, P_n\}$ denote a team of n participants. The TA selects large public prime $p = 2 \cdot q + 1$, where q is also prime. The TA selects a generator (or primitive element) α to $\mathbb{Z}_q$ (see e.g., [8, p. 30]). The basis for the threshold mechanism is Shamir secret sharing scheme [14]. The TA defines the threshold t which conveys the requirement that minimum t participants are required to carry out the threshold computation. For each team, the TA a secret polynomial of order t in agreement with the $(t+1, n)$ threshold requirement:

$$f(x) = \sum_{j=0}^{t} a_j \, x^j$$

Let t coordinates, $\big(j,\ f(j)\big), j \in \{1, \ldots, t\}$, from $f(x)$ constitute the coefficients of a second polynomial $g(x)$:

$$g(x) = \sum_{j=1}^{t} f(j) \, x^{j-1}$$

The TA computes for each $P_i \in \mathcal{U}$ a secret user share

$$s_i = g(i) \pmod{q}$$

in agreement with a (t, n) threshold requirement, and an additional secret user share

$$f_{t+1} = f(t+1) \pmod{q}$$

that is common for all participants of $\mathcal{U}$. The TA computes the public key representing $\mathcal{U}$ as

$$y = \alpha^{-f(0)} \pmod{p}$$

where $f(0) = a_0$.

EPR request. First, the team $\mathcal{U}$ sends a request to the EPR server S to acquire access to a specific EPR denoted m.

Secure EPR transfer. This procedure consists of the following steps:

Step 1. Let y denote the public key of the requesting team. S generates a random secret number r, and subsequently encrypts m according to the ElGamal cryptosystem as

$$c = m \cdot y^r \pmod{p} \quad \text{and} \quad R = \alpha^{-r} \pmod{p}$$

Due to the threshold requirement, a subcoalition $T \subseteq \mathcal{U}$ of at least t team members is required to collaboratively obtain EPR access.

Step 2. Each $P_i \in T$ receives (c, R), computes and broadcasts

$$v_i = R^{s_i} \pmod{p}$$

Step 3. Let $I_T = \{i \mid P_i \in T\}$. After receiving v_j, $j \in I_T$, each participant in T applies Lagrange interpolation on the exponents according to

$$B_i = \prod_{j \in I_T} v_j^{b_j(i)} = \alpha^{-r \cdot f(i)} \pmod{p} \qquad (1 \leq i \leq t)$$

where

$$b_i(x) = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x - j}{i - j} \pmod{q}$$

Each $P_i \in T$, holding the secret f_{t+1}, privately computes

$$B_{t+1} = R^{f_{t+1}} \pmod{p}$$

and subsequently restores the secret encryption factor by Lagrange interpolation

$$Y = \prod_{j=1}^{t+1} B_j^{b'_j(0)} \pmod{p} \quad \text{where} \quad b'_i(x) = \prod_{\substack{j=1 \\ i \neq j}}^{t+1} \frac{x - j}{i - j} \pmod{q}$$

Finally, the participants restore the electronic patient record according to

$$m = c \cdot Y = (m \cdot y^r) \cdot \alpha^{-r\, f(0)} \pmod{p}$$

Since Y can only be computed by means of B_{t+1} by Lagrange interpolation in agreement with the $t+1$ threshold, m can only be restored by the members of $\mathcal{U}$ holding the secret f_{t+1}.

3.2 Security analysis

In this subsection, we present a security analysis to demonstrate that the security of the presented protocol is in agreement with the following security properties. We assume that all computations are in $\mathbb{Z}_p$.

Theorem 1.1. *(Long-term user share confidentiality.)* It is computationally infeasible to obtain secret user shares from user generated messages.

Sketch of proof. During decryption, each participant $P_i \in T$ broadcasts $v_i = R^{s_i} \pmod{p}$. Due to the Discrete Logarithm Problem, it is computationally infeasible to obtain a secret user share s_j from v_i in $\mathbb{Z}_p$. Note that no information is broadcasted that is computed from the secret share f_{t+1}.

Theorem 1.2. *(Secure EPR transfer.)* Only a subcoalition $T \subseteq \mathcal{U}$ of at least t participants can decrypt an encrypted message that is addressed to the given team $\mathcal{U}$.

Sketch of proof. The presented protocol is according to the ElGamal public key cryptosystem [16], where the cryptotext is represented by $c = m \cdot y^{-r}$ and $R = \alpha^r$ where $y = \alpha^{a_0}$, a_0 and r are unknown. The ElGamal cryptosystem is based on the difficulty of finding the secret encryption factor $Y = y^r$. This is equivalent with the difficulty of the Diffie-Hellman Problem which is that given $\alpha^x \pmod{p}$ and $\alpha^y \pmod{p}$, it is computationally infeasible to establish $a^{xy} \pmod{p}$.

Shamir secret sharing is considered to be perfect, meaning that given a (t, n) secret sharing scheme and $t - 1$ or less secret shares are known, no information about the shared secret (i.e., the secret polynomial) can be deduced.

Since the polynomial order of $f(x)$ is t, no information about the secret polynomial is provided given the computations based on t or less shares. The values of $B_j = R^{f(j)} \pmod{p}$, $j \in \{1, \ldots, t\}$, can be publicly computed by means of Lagrange interpolation. Because this is below the threshold $t + 1$ of $f(x)$, it is computationally infeasible to establish $R^{f(j)} \pmod{p}$, $j \notin \{1, \ldots, t\}$, by Lagrange interpolation on the exponents. Thus, only by means of the additional $R^{f(t+i)}$ that is secretly computed by the participants of $T \subseteq \mathcal{U}$, the secret encryption factor $Y = R^{f(0)}$ can be computed and subsequently the EPR restored.

4 Secure acquisition of encrypted medical data

It could be argued that a more overall level of information confidentiality should be preserved, so that confidentiality is not only considered regarding to whom EPR access should be granted, but that medical data could also have a general protection with regard to the long-term storage of such data. For example, it could be pointed out that a medical information system whose medical data is encrypted at a long-term basis may provide an increased level of confidentiality than if the data is stored as plaintext. Nevertheless, this assumes that the corresponding cryptokeys would have to be "out of reach". This could be achieved so that there would be no cryptokey tables, but that such cryptokeys could only be restored as a function of the consensus and the subsequent computations of a minimum number of participants or security administrators. In the remainder of this paper, it is assumed that each EPR is encrypted by a unique and secret key and stored encrypted at the EPR server. There are

no tables of cryptokeys, and the following schemes allows restoration of medical data due to a threshold security requirement. The secret EPR cryptokeys are prevented from being disclosed to the participants.
In this section, we present two complementary broadcast-oriented schemes for secure acquisition of long-term encrypted medical data, whereas the EPR decryption is respectively undertaken at the server side and at the user side. Server-side EPR restoration provides establishment of the secret long-term encryption key at the EPR server for subsequent EPR decryption with a following short-term encryption for secure data distribution to the given medical team. The third scheme requires fewer rounds of broadcasting since EPR restoration is performed at the user-side.
Practical performance considerations show that symmetric secret key encryption has a considerably higher performance than public key encryption. In practical realizations, it would therefore be appropriate to implement symmetric key-based long-term EPR encryption whereas each symmetric key would be encrypted according long-term public key encryption, in contrast to public key-based long-term EPR encryption. Accordingly, user-side EPR decryption would reveal the corresponding long-term secret symmetric keys which could be considered a security weakness. Nevertheless, since both schemes are closely related, it could be of interest to include both.

4.1 Protocol 2. Server-side EPR restoration

The following scheme has two functions: 1) Restoration of the secret EPR cryptokey at the EPR server for subsequent decryption of the pertaining EPR, and 2) subsequent confidential transferral of the pertaining EPR to the team by means of threshold decryption.

Initialization. A trusted authority (TA) is required to set up the scheme, i.e., to provide secret user keys and a public key representing each respective team. Let $\mathcal{U} = \{P_1, P_2, \ldots, P_n\}$ denote a team of n participants. The TA selects large public prime $p = 2 \cdot q + 1$ where q is also prime, and selects a generator (or primitive element) α to $\mathbb{Z}_q$ (see e.g., [8, p. 30]).
The basis for the threshold mechanism is the Shamir secret sharing scheme. The TA defines the threshold t which conveys the requirement that minimum t participants are required to carry out the threshold computation. For each team, the TA a secret polynomial of order t corresponding to a $(t+1, n)$ threshold requirement:

$$f(x) = \sum_{j=0}^{t} a_j \, x^j$$

where a_j is selected randomly such that $\beta = f(1) = \sum_{j=0}^{t} a_j$ is equal for all teams. Let t coordinates $\big(j,\ f(j)\big)$, $j \in \{1, \ldots, t\}$, from $f(x)$ constitute

the coefficients of $g(x)$:

$$g(x) = \sum_{j=1}^{t} f(j)\, x^{j-1}$$

The TA computes for each $P_i \in \mathcal{U}$ a secret user share

$$s_i = g(i) \pmod{q}$$

in agreement with a (t, n) threshold requirement, and an additional secret user share

$$f_{t+1} = f(t+1) \pmod{q}$$

that is shared by all participants of $\mathcal{U}$. The TA computes the public key that represents $\mathcal{U}$ as

$$y = \alpha^{-f(0)} \pmod{p}$$

where $f(0) = a_0$.
Each patient is represented by a unique identity id where each $id \neq \alpha$. The EPR m of patient id is encrypted according to the ElGamal public key cryptosystem [16], and stored as cryptotext on the EPR server due to

$$z = m \cdot w^{-1} \pmod{p}$$

where $w = id^{\beta \cdot k} \pmod{p}$ is a long-term secret EPR encryption factor, and k is a long-term secret key of S.

EPR request. First, the team $\mathcal{U}$ sends a request (id, y) to the EPR server S to acquire access to a specific EPR denoted m. Let y denote the public key of the requesting team and id the identity of the patient whose EPR is requested.

Secure server-side EPR restoration.
Step 1. S generates two random secret numbers r_1 and r_2 from $\mathbb{Z}_p$, and computes and broadcasts the challenge

$$c = id^k \cdot \alpha^{r_1 \cdot r_2} \pmod{p} \quad \text{and} \quad R = \alpha^{-r_1} \pmod{p}$$

Step 2. The team members receives (c, R). Each $P_i \in T$ computes and broadcasts

$$u_i = c^{s_i} \pmod{p} \quad \text{and} \quad v_i = R^{s_i} \pmod{p}$$

Step 3. Since only S holds the secret r_2, S exclusively restores the secret encryption factor according to Lagrange interpolation

$$\begin{aligned} w &= \prod_{i \in I_T} (u_i \cdot v_i^{r_2})^{b_i(1)} \pmod{p} \\ &= (id^{k \cdot f(1)} \cdot \alpha^{r_1 \cdot r_2 \cdot f(1)}) \cdot (\alpha^{-r_1 \cdot r_2 \cdot f(1)}) = id^{k \cdot \beta} \pmod{p} \end{aligned}$$

where

$$b_i(x) = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x-j}{i-j} \pmod{q}$$

Secure EPR transfer. S computes and broadcasts

$$A = z \cdot w \cdot y^{r_1} = m \cdot y^{r_1} \pmod{p}$$

Each participant in T applies Lagrange interpolation on the exponents according to

$$B_i = \prod_{j \in I_T} v_j^{b_j(i)} = \alpha^{-r_1 \cdot f(i)} \pmod{p} \qquad (1 \leq i \leq t)$$

Each participant, holding the secret number f_{t+1}, privately computes

$$B_{t+1} = R^{f_{t+1}} \pmod{p}$$

and subsequently restores the secret encryption factor by Lagrange interpolation

$$Y = \prod_{j=1}^{t+1} B_j^{b'_j(0)} \pmod{p} \quad \text{where} \quad b'_i(x) = \prod_{\substack{j=1 \\ i \neq j}}^{t+1} \frac{x-j}{i-j} \pmod{q}$$

in order to restore the electronic patient record m according to

$$m = A \cdot Y = (m \cdot y^{r_1}) \cdot (\alpha^{-r_1 \cdot f(0)}) \pmod{p}$$

Note that since Y can only be computed by means of B_{t+1} by Lagrange interpolation in agreement with the $t+1$ threshold, m can only be restored by the members of $\mathcal{U}$ holding the secret f_{t+1}.

4.2 Security analysis

In this subsection, we present a security analysis to demonstrate that the security of the presented protocol is in agreement with the following security properties. We assume that all computations are in $\mathbb{Z}_p$.

Theorem 2.1. *(Long-term user share confidentiality.)* It is computationally infeasible to obtain secret user shares from user generated messages. See the security analysis for Theorem 1.1. for sketch of proof.

Theorem 2.2. *(Confidentiality of the long-term encryption factor.)* (Secure server side EPR restoration.) The protocol prevents that the secret long-term encryption factor w to any other than S.
Sketch of proof. Due to the threshold requirement, the contributions (i.e., the partial decryptions) from a subcoalition $T \subseteq \mathcal{U}$ of at least t participants are required to establish the secret encryption factor w.

The first broadcast of $c = id^k \cdot \alpha^{r_1 \cdot r_2}$ and $R = \alpha^{-r_1}$ where (r_1, r_2) are secret, is equivalent to the ElGamal cryptosystem [16], and thus the secret factor id^k is protected.
After Step 3, the numbers $(id^{k \cdot \beta} \cdot \alpha^{r_1 \cdot r_2 \cdot f(1)})$ and $\alpha^{-r_1 \cdot f(1)}$ can be publicly established by means of Lagrange interpolation. However, due to the ElGamal cryptosystem, the encryption factor $w = id^{k \cdot f(1)}$ can only be restored by means of r_2, which is known by S only. Thus, it is prevented that w is revealed to other than S.

Theorem 2.3. *(Secure EPR transfer.)* Only a subcoalition $T \subseteq \mathcal{U}$ of at least t participants can decrypt an encrypted message that is addressed to the given team $\mathcal{U}$.
See the security analysis for Theorem 2.1 for sketch of proof.

4.3 Protocol 3. User-side EPR restoration

This scheme is more efficient than the previous since both the restoration (decryption of the long-term encrypted EPR) and secure transfer of the EPR are done in the same step.

Initialization. Same as Protocol 2. Recall that $z = m \cdot w^{-1} \pmod p$ and $w = id^{\beta \cdot k} \pmod p$. Additionally, $K = \alpha^{\beta} \pmod p$ is a public parameter.

EPR request. First, the team $\mathcal{U}$ sends a request to the EPR server S to acquire access to a specific EPR denoted m.

Secure EPR transfer and user-side restoration.
Step 1. S generates two random secret numbers r_1 and r_2 from $\mathbb{Z}_p$, and then computes and broadcasts (c, d, e) where

$$\begin{aligned} c &= z \cdot K^{r_1} \pmod p \\ d &= id^k \cdot \alpha^{-r_1} \cdot y^{r_2} \pmod p \\ e &= K^{-r_2} \pmod p \end{aligned}$$

Due to the public key y, this can only be handled by the pertaining team $\mathcal{U}$.

Step 2. After receiving (c, d, e), each $P_i \in T$ computes and broadcasts

$$u_i = d^{s_i} \pmod p \quad \text{and} \quad v_i = e^{s_i} \pmod p$$

Step 3. After receiving the computational results from the other participants in Step 2, each participant in T computes by Lagrange interpolation

$$A = \prod_{i \in I_T} u_i^{b_j(1)} = d^{\beta} \pmod p$$

and

$$B_i = \prod_{j \in I_T} v_j^{b_j(i)} = \alpha^{-\beta \cdot r_2 \cdot f(i)} \pmod p \quad \text{where} \quad b_i(x) = \prod_{\substack{j \in I_T \\ i \neq j}} \frac{x - j}{i - j} \pmod q$$

for $i \in \{1, \ldots, t\}$. Each participant in T, holding the secret number f_{t+1}, privately computes

$$B_{t+1} = e^{f_{t+1}} \pmod{p}$$

and computes by Lagrange interpolation

$$C = \prod_{j=1}^{t+1} B_j^{b'_j(0)} = \alpha^{-\beta \cdot r_2 \cdot f(0)} = y^{-\beta \cdot r_2} \pmod{p} \quad \text{where} \quad b'_i(x) = \prod_{\substack{j=1 \\ i \neq j}}^{t+1} \frac{x-j}{i-j} \pmod{q}$$

and then privately restores the EPR according to

$$m = c \cdot A \cdot C \pmod{p}$$

i.e., $m = (m \cdot id^{-\beta \cdot k} \cdot \alpha^{\beta \cdot r_1}) \cdot (id^{k \cdot \beta} \cdot y^{r_2 \cdot \beta} \cdot \alpha^{-r_1 \cdot \beta}) \cdot (y^{-\beta \cdot r_2})$. Note that m can only be restored by means of C in agreement with the $t+1$ threshold. This can only be computed by the members of $\mathcal{U}$ only, holding the secret share f_{t+1}.

4.4 Security analysis

In this subsection, we present a security analysis to demonstrate that the security of the presented protocol is in agreement with the following security properties. We assume that all computations are in $\mathbb{Z}_p$.

Theorem 3.1. *(Preservation of the confidentiality of long-term server-side data.)* It is computationally infeasible to obtain w and id^k.
Sketch of proof. An essential security property is to preserve the secrecy of the secret long-term encryption factor w and id^k also after that m has been restored at the user side. S broadcasts

$$\begin{aligned} c &= z \cdot K^{r_1} = (m \cdot w^{-1}) \cdot (\alpha^{\beta})^{r_1} \\ d &= id^k \cdot \alpha^{-r_1} \cdot y^{r_2} \\ e &= K^{-r_2} \end{aligned}$$

where r_1 and r_2 are secretly known by S.
Due to the Discrete Logarithm Problem, it is computationally infeasible to obtain r_2 given e.
Regarding c, the long-term secret encryption factor w is protected by the secret factor K^{r_1}, even if m is revealed. Regarding d, the secret factor id^k is protected by the secret factors α^{-r_1} and y^{r_2}. Note that the secret r_1 cryptographically binds c and d, and the secret r_2 cryptographically binds e and d. We see that all the factors of (c, d, e) are distinct.
The factor K^{r_1} must be known to deduce w. K^{r_1} could be computed from α^{r_1} which is a secret factor of d. However, this is protected by the secret y^{r_2}.

Theorem 3.2. *(Secure EPR transfer.)* Only a subcoalition $T \subseteq \mathcal{U}$ of minimum t participants can obtain m given (c, d, e).

Sketch of proof. The cryptotext is represented as (c, d, e). However, multiplying c and d gives $c' = m \cdot y^{r_2 \cdot \beta}$. The cryptogram $(c',\ e)$ (where $e = \alpha^{\beta \cdot r_2}$) is in full agreement with the ElGamal public key cryptosystem [16]. Thus, the team participants are required to collaboratively compute $C = e^{f(0)}$ in agreement with the threshold requirement to establish m. See Theorem 1.2. for more on the threshold issue.
As noted, preserving the secrecy of w and id^k is an essential security property. However, if the factors w and id^k would become compromised, we get by multiplication, respectively $c \cdot w = m \cdot K^{r_1}$, and $d \cdot id^{-k} = \alpha^{-r_1} \cdot y^{r_2}$. These two results contain both two unknown factors which cannot further be disclosed. Thus, it is computationally infeasible to obtain m given (c, d, e, w, id^k).

5 Conclusion

In this paper, we have focused on the importance of preserving the confidentiality of personal medical data. We have pointed out that legitimacy should be a main criteria for medical professionals to obtain access to medical patient data, and that the consensus of a minimum number of associated practitioners in medical teams is a reasonable basis for such practitioners to acquire access to medical data.
In this context, we have presented three closely related cryptographic protocols for secure team-based EPR access acquisition over wireless networks. The first is basically a novel broadcast-oriented threshold-decryption scheme enabling the EPR server to securely transfer medical data to medical teams. The data can only be decrypted and obtained if the number of active team members is in agreement with the threshold requirement. For the remaining two protocols, it is assumed that the EPRs are stored encrypted at the EPR server, and that each is encrypted with a unique secret key. There are no cryptokey tables, but the protocols enable medical teams to acquire and restore the requested EPR without that the pertaining cryptokey is revealed. The second protocol provides decryption of the EPRs at the server side while the third provides decryption at the user side. All schemes are secure in the sense that they provide secure team-based acquisition and secure transfer of medical data.
Future works could be to consider team-based EPR access acquisition for ad-hoc teams, i.e., temporary coalitions of medical practitioners that are not issued a long-term public key in advance.

References

[1] American Medical Association. Patient Confidentiality. See http://www.ama-assn.org/ama/pub/category/4610.html.

[2] T. Rindfleich. Privacy, Information Technology and Health Care. Communications of the ACM, 1997, Vol. 40 (8).

[3] J. Biskup and G. Bleumer. Cryptographic protection of health information: cost and benefit. International Journal of Bio-Medical Computing, 43:61–67, 1996.

[4] M. Eichelberg et. al. A survey and analysis of Electronic Healthcare Record standards. ACM Computing Surveys, Vol. 37(4), 2005.

[5] G . Serour. Confidentiality, privacy and security of patients' health care information: FIGO Committee for the Ethical Aspects of Human Reproduction and Women's Health. International Journal of Gynecology & Obstetrics, Vol. 93, Iss. 2, pp. 189-190, 2006.

[6] J. Bergmann, O. Bott, D. Pretschner, and R. Haux. An e-consent-based shared EHR system architecture for integrated healthcare networks. International Journal of Medical Informatics, 76(2-3):130 – 136, 2007.

[7] S. Eskeland, V. Oleshchuk. EPR Access Authorization of Medical Teams Basded on Patient Consent. 2nd European Conference on eHealth, 2007.

[8] J. Pieprzyk, T. Hardjono, J. Seberry. Fundamentals of computer security. ISBN 3-540-43101-1, Springer-Verlag, 2003.

[9] Y. Desmedt, Y. Frankel. Threshold cryptosystems. Advances in Cryptology, Proc. of Crypto'89, LNCS, pp. 307–315, Springer-Verlag, 1990.

[10] T. Pedersen. A threshold cryptosystem without a trusted party (Extended Abstract). Eurocrypt '91, LNCS, Vol. 547, pp. 522–526, Springer-Verlag, 1991.

[11] S. Saeednia, H. Ghodosi. A Self-Certified Group-Oriented Cryptosystem Without a Combiner. LNCS, Vol. 1587, pp. 192–201, Springer-Verlag, 1999.

[12] L. Harn. Group-oriented (t, n) threshold digital signature scheme and digital multisignature. Computers and Digital Techniques, IEE Proceedings. Vol. 141, No. 5, pp. 307-313, 1994.

[13] C. M. Li, T. Hwang, N. Y. Lee. Threshold-Multisignature Schemes where Suspected Forgery Implies Traceability of Adversarial Shareholders, Eurocrypt 1994, pp. 194–204.

[14] A. Shamir. How to Share a Secret. Communications of the ACM, Vol. 22, No. 11, pp. 612–613, 1979.

[15] J. Pieprzyk, C. H. Li. Multiparty key agreement protocols. IEE Proceedings, Computer and Digital Techniques, Vol. 147, No. 4, pp. 229–236, 2000.

[16] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469–472, 1985.

Paper I

A Decentralized Hierarchical Access Control Scheme for the Medical Scenario

Sigurd Eskeland and Neeli Prasad

A Decentralized Hierarchical Access Control Scheme for the Medical Scenario

Sigurd Eskeland
University of Agder
Grooseveien 36
N-4876 Grimstad, Norway
sigurd.eskeland@uia.no

Neeli Prasad
Center for TeleInFrastructur (CTIF), Aalborg University
Niels Jernes Vej 12
DK-9220 Aalborg, Denmark
np@kom.auc.dk

Abstract
Electronic patient records contain highly personal and confidential information that it is essential to keep private. Thus, only the medical professionals providing care to a patient should access the patient record of the concerning patient. As personal medical data can be considered to be the property of the corresponding patient, it is justified that patients should have the opportunity to exert control over their own data. In this paper, we propose a cryptographic access control scheme allowing patients to grant medical teams authorizations to access their medical data. Moreover, the hierarchical aspects of teams are taken into account so that the modules of the patient record are to be accessed according to the individual privileges of the medical professionals of the team. Thus, more privileged users obtain larger portions of the data than less privileged users.

1 Introduction

With the increased application of information technology in health care, there has been extensive focus on the security issues of electronic patient records (EPR) in medical environments [1, 2]. These issues include the pushing need for protecting personal medical data. Many security systems are based on predefined and preassigned credentials or permissions like roles or entries in access control lists, issued by an authority [3, 4]. This works fine in environments characterized by relatively fixed work tasks, and where changes in assignments of permissions and authorizations happen relatively infrequently. In contrast, medical environments are characterized by being highly dynamical, incorporating a high number of professionals providing care to temporary hospitalized patients. The medical scenario would normally require an active security administrator granting EPR access to legitimate medical professionals on behalf

of the patients, which could be a demanding task due to complexity and workload. A major issue here is that all patient records must be managed properly and confidentially with respect to the private and sensitive information they hold.

Due to the confidential nature of personal medical data, such information should only be disclosed to legitimate medical personnel, i.e, the EPR of a patient should only be disclosed to those medical practitioners that are currently providing care to that patient. Since EPRs contain the personal medical data of patients, they can be considered to be the property of the patients, and it could therefore be justified that patients should have the opportunity to exert control over their own data. This could be realized by providing patients the privilege to grant medical personnel access to their own medical data by consent, and thereby the privilege of controlling whom is to access their data. Access control schemes for the medical scenario that include the patient as a consenting and granting entity seem, however, to be virtually absent in the literature.

Medical teams involves not only the multi-party aspect but also the hierarchical aspect due to that various types of medical practitioners have different rankings. For instance, doctors are ranked above nurses. In this context, it is reasonable or even necessary that practitioners of lower rankings should be privileged access to less sensitive or confidential medical information than those of higher rankings.

The contribution of this paper is two-fold. Firstly, we present a decentralized hierarchical key multi-party agreement protocol. By decentralized we mean that the protocol is contributory, i.e., that each participant contributes to the values of the established secret keys in contrast to centralized key transfer. Secondly, it includes an EPR granting step where the patient acknowledges to the EPR server whether or not the given team should be authorized access to the patient's EPR. Successful completion of this stage signifies hence that the medical data of interest will be securely shared with each of the corresponding team members, and the data is then subsequently encrypted using the newly computed hierarchical session keys and transferred to the team.

The hierarchical key agreement protocol provides secure establishment of a hierarchy of session keys in agreement with the privilege levels of the team members. The hierarchical key agreement protocol allows team members to obtain the session keys of their own and underlying security levels while the opposite is prevented, thus ensuring that the personnel individually get access to data in agreement with their ranking and privileges.

The scheme assumes consequently that the EPRs consists of data modules that are each assigned a confidentiality/sensistivty level reflecting the sensitivity level of its medical data. The EPR sensitivity hierarchy is hence in agreement with the user hierarchy. Accordingly, this facilitates that each medical participant is to only receive access to medial data that are in agreement with his or her ranking.

After the patient has successfully verified and consented to the composition and authenticity of the members of the pertaining medical team, the EPR server encrypts the EPR modules by means of the corresponding hierarchical session keys in agreement with the confidentiality level of each respective EPR module, and then transfers the encrypted medical data to the pertaining medical team. Hence, secure access control and secure transport of the data over insecure networks are provided.

2 Related work

Hierarchical access control (HAC) is a class of cryptographic schemes that supports establishment and deduction of long-term, predefined cryptographic cryptokeys in agreement with a predefined user hierarchy. Users of a given security class are able to securely compute such keys associated with their own and underlying security classes, while computation of keys associated with overlying security classes is prevented. In contrast, the presented scheme provides secure establishment of a hierarchy of "fresh" sessions keys for ad-hoc hierarchical groups. Computation of hierarchical predefined keys and not hierarchical sessions keys is reasonably a considerable limitation of the applicability and usefulness of such schemes. Many HAC schemes are compliant with user dynamics, i.e., inclusion and exclusion of users and corresponding renewal of hierarchical keys for the pertaining security classes. Examples of HAC schemes can be found in [5–8].
Regarding tree-based key management schemes, the users are logically organized according tree structures, where they are represented as leaf nodes of a binary tree. Due to the to tree structure, this allows them to obtain a common key that is at the root. Thus, such schemes are not hierarchical due to that the users obtain one shared secret key. Examples are Tree-based Group Diffie-Hellman agreement [11], and others in [9,10].

3 Preliminaries

3.1 Security requirements

The proposed hierarchical key establishment protocol is based on the following cryptographic security requirements:

- The protocol must be secure against passive attacks. Only legitimate users must be able to establish secret hierarchical session keys. Messages of former sessions and old session keys cannot be used to deduce future session keys.
- The protocol must be secure against substitution attacks. Each participant must be authenticated and cryptographically linked to the current session certifying the identity and the assigned hierarchical level of the participant.

- Onewayness. The participants of each level can compute the same hierarchical session key and session keys of lower security levels, while computation of hierarchical session keys of higher security levels must be prevented. The user authentication must thus verify that the alleged security level of each user is legitimate, as well as the user himself.
- Forward secrecy. Compromise of long-term secret user keys must not reveal former hierarchical session keys.
- Key freshness. It must be guaranteed that hierarchical session keys are unique and "fresh" for each run of the protocol. Thus, the protocol must be contributory so that all participants contribute equally to the values of the hierarchical session keys of their respective security level.

The security requirements must be satisfied in presence of passive adversaries possessing these capabilities. An active adversary may be able to modify (i.e., add, replace, replay, change) any broadcasted messages. Attempt to impersonate any legitimate user by replaying old messages must be infeasible.

3.2 Hierarchical preliminaries

The participants are denoted $\mathcal{U} = \{P_1, \ldots, P_m\}$, where P_1 represents the patient, $\{P_2, \ldots, P_{m-1}\}$ represent the members of the medical team, and P_m represents the EPR server. Each participant $P_i \in \mathcal{U}$ is associated with one hierarchical user class (or security class) S_ℓ, i.e., $P_i \in S_\ell$. Let L_i denote the security level of P_i. Each level ℓ contains one security class

$$S_\ell = \{P_j \,|\, P_j \in \mathcal{U} \quad \text{and} \quad \ell = L_j\}$$

where $1 \leq \ell \leq \lambda$, that includes all participants of that level.
Let $\mathcal{U} = \{S_1, \ldots, S_\lambda\}$ be divided into λ disjoint hierarchical security classes where S_λ denotes the top security class. The security classes are hierarchically ranked according to the relation $\prec$ so that $S_i \prec S_j$ if $i < j$. This indicates that S_j has a higher ranking than S_i.

4 The protocol

The contribution of this paper is two-fold. Firstly, it constitutes the hierarchical key establishment protocol involving the team, server and patient which is executed first. Secondly, it includes an EPR granting step where the patient acknowledges to the EPR server whether or not the given team should be authorized access to the patient's EPR. Successful completion of this stage signifies hence that the medical data of interest will be securely shared with each of the corresponding team members, and the data is then subsequently encrypted using the newly computed hierarchical session keys and transferred to the team.

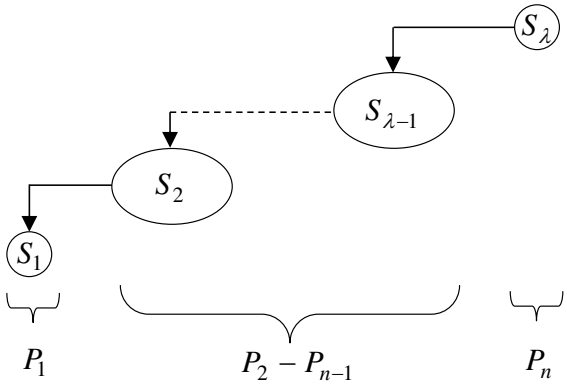

Figure 1: Hierarchical key deduction

The presented protocol authenticates each user and their ranking. A compromised user arrangement where someone pretends to have a higher ranking than his real ranking will be detected by signature certification. The protocol is based on the scheme presented in [14].

4.1 User arrangement

In agreement with Section 3.2, the users are arranged in increasing order according to their ranking. We moreover assume that the users are linearly ordered so that all participants of $\mathcal{U}$ are organized sequentially, forming a logical line, where $P_1 \in S_1$ (the patient) is sequentially first and $P_m \in S_\lambda$ (the EPR server) is sequentially last and hierarchically topmost. The users P_i and P_{1+1}, $1 \leq i < m$, are an adjacent pair of users, regardless of whether they are in the same security class or in adjacent security classes. The hierarchical arrangement is represented in Fig. 1.
The sequential ordering of the users implies that participants positioned first and last in a given level are adjacent with a participant of the respective adjacent underlying and overlying security levels (if any).
The participants could, for instance, be ordered within each security level according to identity. To ensure a different user order for each session, the participants could be ordered according to $f(ID_i, T)$ within each security level, where f is a secure one-way-function and T is the current time. A change in the hierarchy, inclusion of new participants or participants leaving, requires protocol rerun.

4.2 The hierarchical key agreement protocol

It is assumed that all participants in $\mathcal{U}$ are arranged with respect to increasing privilege level as previously described.

Initializations. A large public prime number p is selected where $p = 2 \cdot q + 1$ and q is also a prime. Let α be a public generator to $\mathbb{Z}_q$ (e.g., see [13, p.

30]). This can be done by an offline trusted third party (TTP) that also sets up the certification and signature scheme.

Round 1. Each user $P_j \in \mathcal{U}$, generates a random number $r \in \mathbb{Z}_q$ and computes and broadcasts the ephemeral DH public key $z_j = \alpha^{r_j} \pmod p$.

Round 2. Each user P_j, $2 \leq j \leq m-1$, establishes two secret DH keys, $k_{j-1,j} = z_{j-1}^{r_j} \pmod p$ and $k_{j,j+1} = z_{j+1}^{r_j} \pmod p$, with each adjacent participant, respectively.
A participant is sequentially positioned first in S_ℓ if $\ell = L_{i-1} + 1 = L_i$. The sequentially first participant of each security class computes

$$v_j = \widehat{k}_{j-1,j} - f(k_{j,j+1}) \pmod p$$

where f is a secure one-way function. The hat in $\widehat{k}_{j-1,j}$ denotes it is an "entrance key" whereof knowledge enables participants to deduce hierarchical session keys of underlying levels.
The application of the one-way function in the last term of v_j enforces the hierarchical one-way security property, preventing participants to deduce hierarchical session keys of overlying security levels, while the converse is permitted.
Subsequently positioned participants $P_j \in S_\ell$, compute

$$v_j = k_{j-1,j} - k_{j,j+1} \pmod p$$

The patient (P_1) and the EPR server (P_m) who do not have two adjacent users, compute an arbitrary value, say, $v_1 = v_m = c$ where $c = f(z_1, \ldots, z_m)$ represents the current session itself and f is a secure one-way-function. All users $P_j \in \mathcal{U}$ compute a signature $Sig_j(v_j, c)$ that binds the user (identity and privilege level) with v_j and the current session itself c. To complete the second round, each user $P_j \in \mathcal{U}$ broadcasts $\big(ID_j, v_j, Sig_j(v_j, c)\big)$. It is assumed that the user privilege level is cryptographically incorporated into the user identity or user certificate.

Authentication and hierarchical session key computation. Each user $P_j \in \mathcal{U}$ certifies the signature for the other participants, and makes sure that the hierarchical user arrangement is not compromised. It must moreover be verified that the "entrance keys" are computed according to the protocol. To do this, first compute a candidate entrance key $k'_{j-1,j} = v_j + f(k_{j-1,j})$. Now it can be verified if

$$v_j^2 \stackrel{?}{\equiv} k'^2_{j-1,j} - 2\,k'_{j-1,j}\,f(k_{j,j+1}) + f(k_{j,j+1})^2 \pmod p$$

holds. This verification will detect whether the participant has cheated. Each user $P_i \in S_\ell$ is now able to compute the hierarchical session key

$$\begin{aligned} K_\ell &= (k-j) \cdot z_{j-1}^{r_j} + \sum_{l=i+1}^{j-1} (l-j) \cdot v_l - \sum_{l=k+1}^{j} (k-l) \cdot v_l \\ &= k_{i,i+1} + k_{i+1,i+2} + \ldots + k_{k-2,k-1} + k_{k-1,k} \pmod p \end{aligned}$$

The indices i and k refer respectively to the sequentially first and last user indices of S_ℓ, where $\ell = L_{i+1} + 1 = L_i = L_k = L_{k+1} + 1$.
The secure one-way-function could be a cryptographic hash function with byte numbers that conform in size with p, or exponentiation by for a given $\alpha^{k_{j-1,j}} \pmod p$ due to the Discrete Logarithm Problem.

Computing lower-level hierarchical session keys. A user $P_k \in S_\ell$ deduce lower-level session keys by first computing the entrance key for the underlying level as

$$\widehat{k}_{i-1,i} = v_i + f(k_{j-1,j}) \pmod p$$

where $k_{j-1,j} = \sum_{j=i+1}^{k-1} v_j + k_{k-1,k} \pmod p$ for the subordinate security class $S_{\ell-1}$. For each subsequent class S_δ between target S_γ and S_ℓ, $\gamma < \delta$, iteratively compute the entrance key of $S_{\delta-1}$ where $k_{k-1,k}$ refers to the entrance key of S_δ. The target session key is finally computed as

$$\begin{aligned} K_\gamma &= (k-1) \cdot \widehat{k}_{k-1,k} + \sum_{l=i+1}^{k-1} (k-l) \cdot v_l \pmod p \\ &= k_{i,i+1} + k_{i+1,i+2} + \ldots + k_{k-2,k-1} + k_{k-1,k} \pmod p \end{aligned}$$

where i and k refer to the start and end user indices of S_γ.

4.3 Validation and granting

The preceding subsections describe the arrangements for hierarchical key computation. It is upon this basis that the patient (P_1) will be able to certifiably evaluate the composition of the concerning team and thereby approve authorization for EPR access. The server (P_m) obtains $k_{1,2}$ according to the alleged (but the not yet validated by the patient) composition of the team. The server (P_m) then signs the hash of $k_{1,2}$, a list of the alleged identities of the team (ID) and a random nonce N_m, and sends this to the patient (P_1). The patient (P_1) certifies the signature with the public key of the server (P_m). If valid, the patient checks the team list (ID), and if the patient (P_1) consents to that the team should be authorized access to his EPR, he or she signs the nonce and the (non-secret) $k_{1,2}$, and returns the result to the server. Otherwise, he or she aborts. The server (P_m) certifies the message of the patient (P_1) by means of the patient's public key and the value of $f(k_{1,2})$. The team validation and granting is as follows:

1. $P_m \rightarrow P_1$: N_m, $Sig_m\big(ID,\, f(k_{1,2}),\, N_m\big)$
2. $P_1 \rightarrow P_m$: $Sig_1\big(f(k_{1,2}),\, N_m\big)$

In the last step, if the signature is verifiably correct, the EPR server provides the team access to the given EPR, encrypted by the secret hierarchical session keys.

4.4 Example

To illustrate the scheme, in this example eight users are located in two security levels, $S_1 = \{P_1, P_2, P_3, P_4\}$ and $S_2 = \{P_5, P_6, P_7, P_8\}$ where $S_1 \prec S_2$. Now $P_8 \in S_2$ can compute K_2 according to

$$\begin{aligned} K_2 &= v_6 + 2\,v_7 + 3\,k_{7,8} \\ &= (k_{5,6} - k_{6,7}) + 2\,(k_{6,7} - k_{7,8}) + 3\,k_{7,8} \pmod{p} \\ &= k_{5,6} + k_{6,7} + k_{7,8} \pmod{p} \end{aligned}$$

Next, $P_8 \in S_2$ derives the entrance key $\widehat{k}_{4,5}$ according to

$$\widehat{k}_{4,5} = v_5 + f(v_6 + v_7 + k_{7,8} \bmod p) \pmod{p}$$

Finally, he or she computes

$$\begin{aligned} K_1 &= v_2 + 2\,v_3 + 3\,(v_4 + \widehat{k}_{4,5}) \pmod{p} \\ &= k_{1,2} + k_{2,3} + k_{3,4} \pmod{p} \end{aligned}$$

5 Security analysis

In this section, we show that the scheme is secure in agreement with the security requirements presented in Section 3.1.

Security Requirement 1. The protocol must be secure against passive attacks.
Proof. According to the computational DH assumption, it is computationally infeasible to compute any $k_{i-1,i}$ for $1 < i \leq m$ after Round 1 without the secret r_{i-1} or r_i. In Round 2; given $k_{i-1,i} - k_{i,i+1}$, no information is revealed about $k_{i-1,i}$ and $k_{i,i+1}$. Since hierarchical session keys are established by knowledge of the secret $k_{i-1,i}$ for a given level, it is computationally infeasible to passively obtain any hierarchical session keys.

Security Requirement 2. The protocol must be secure against substitution attacks.
Proof. A given session is represented by the value of c. The individual contributions of each participant in Round 2 are v_i. The signature of $P_i \in \mathcal{U}$ cryptographically links c and to his or her public key. Assuming the signature scheme is secure, and that the participants' private long-term keys are secret, it is computationally infeasible for an adversary to compute a forged signature for an illegitimate v_i' and the session c. The adversary cannot reuse old signatures that are linked to corresponding former values of v_i and c, because the adversary can neither control the secret DH keys due to the contributory nature of DH key agreement, and thus not the

values of v_i and c, since he cannot substitute the ephemeral DH keys of the other participants without being detected, due to signatures.
If the adversary reuses a former v_i (and thereby corrupting deduction of the hierarchical session keys), along with the concerning signature, the fraud will be detected by certifying the signatures, since it does not match the current c. Thus, because the signature scheme prevents substitution attacks, the adversary cannot succeed reusing old messages and signatures.

Security Requirement 3. The protocol must provide the hierarchical one-way security property.
Proof. Due to $v_i = k_{i-1,i} - f(k_{i,i+1})$, it is computationally infeasible to obtain $k_{i,i+1}$ if f is secure.
If $P_i^{(1)} \in S_\ell$ attempts to violate the hierarchical one-way security property by broadcasting $v_i = k_{i-1,i} - k_{i,i+1}$ instead of $v_i = k_{i-1,i} - f(k_{i,i+1})$, this would be detected by verifying

$$v_j^2 \stackrel{?}{\equiv} k'^2_{j-1,j} - 2\,k'_{j-1,j}\,f(k_{j,j+1}) + f(k_{j,j+1})^2 \pmod{p}$$

for a candidate entrance key $k'_{j-1,j}$. If the verification fail, such a violation is detected and the protocol therefore aborts.

Security Requirement 4. The protocol must provide forward secrecy. Forward secrecy is defined in [12, p. 50] as compromise of a long-term user key will not lead to compromise of session keys that were previously established by means of that long-term user key.
Proof. The values of established keys depend solely on the random number generated by the participants for each session. I.e., the long-term user keys do not influence the values of the hierarchical keys. Only the broadcasted key establishment messages, v_i and c, are signed. Thus, the signatures contain no other hierarchical key information than the already broadcasted key establishment messages. Assuming that a secure signature scheme is employed (Assumption 2); if a corresponding long-term user key is known, it is infeasible to deduce any information of $k_{i-1,i}$, $k_{i,i+1}$ and r_i from the corresponding signature $Sig_i(v_i, c)$. Forward secrecy is therefore achieved.

Security Requirement 5. The protocol must provide hierarchical session keys that are unique for each session.
Proof. Each user in each security level S_ℓ contributes equally to the values of the corresponding hierarchical key K_ℓ. According to Security Requirement 2, employment of a secure signature scheme withstands substitution attacks. Thus, no single user can enforce old keys to be reestablished. Thus, key freshness is ensured.

6 Conclusion

In this paper, we have addressed that since personal medical data can be considered to be the property of patients, it is justified that patients should have the opportunity to exert control over their own data. We have proposed a cryptographic access control scheme allowing patients to grant medical teams authorizations to access their respective electronic patient records. Moreover, the hierarchical aspects of teams are taken into account so that the modules of the patient record are to be accessed according to the individual privileges of the medical professionals of the team. Thus, more privileged users obtain larger portions of the data than less privileged users. The hierarchical key establishment protocol is well-suited for wireless networks due to the broadcasting, and is highly efficient due to only two rounds.

References

[1] T. Rindfleich. Privacy, Information Technology and Health Care. Communications of the ACM, 1997, Vol. 40 (8).

[2] M. Eichelberg et. al. A survey and analysis of Electronic Healthcare Record standards. ACM Computing Surveys, Vol. 37(4), 2005.

[3] W. Talone et. al. Access Control in Collaborative Systems. ACM Computing Surveys, Vol. 37, No. 1, 2005, pp. 29 – 41.

[4] R. S. Sandhu, E. J. Coyne, C. E. Youman. Role-based Access Control Models. IEEE Computer, vol. 29 (1996), pp. 40 - 48.

[5] F. Kuo, V. Shen, T. Chen, F. Lai. Cryptographic key assignment scheme for dynamic access control in a user hierarchy. IEE Proc. Computers & Digital Techniques, Vol. 146, No. 5, 1999, pp. 235 – 240.

[6] C. Lin. Dynamic key management schemes for access control in a hierarchy. Computer communications, Vol. 20, No. 15, pp. 1381 – 1385, 1997.

[7] C. Chang, C. Lin, W. Lee, P. Hwang. Secret sharing with access structures in a hierarchy. AINA, Vol. 2, pp. 31 – 34, 2004.

[8] X. Zou, B. Ramamurthy, S. Magliveras. Chinese Remainder Theorem based hierarchical access control for secure group communications. ICICS, LNCS, Vol. 2229, pp. 381 – 385, 2001.

[9] L. Dondeti, S. Mukherjee, A. Samal. DISEC: a distributed framework for scalable secure many-to-many communication. Proc. of 5th IEEE ISCC, pp. 693 – 698, 2000.

[10] A. Sherman, D. McGrew. Key establishment in large dynamic groups using one-way fucntion tree. IEEE transactions on Software Engineering, Vol. 29, No. 5, pp. 444 – 458, 2003.

[11] Y. Kim, A. Perrig, G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. Proc. of 7th ACM CCS, pp. 235 – 244, 2000.

[12] C. Boyd, A. Mathuria. Protocols for Authentication and Key Establishment. ISBN 3-540-43107-1, Springer-Verlag, 2003.

[13] J. Pieprzyk, T. Hardjono, J. Seberry. Fundamentals of computer security. ISBN 3-540-43101-1, Springer-Verlag, 2003.

[14] S. Eskeland. Efficient Hierarchical Conference Key Establishment in Wireless Networks. IASTED International Conference on Communication, Network and Information Security '05, pp. 94 – 98, Acta Press, 2005.

[15] M. Burmester, Y. Desmedt. A secure and efficient conference key distribution system. In proc. of Eurocrypt'94, LNCS, vol. 950, pp. 275 – 286, Springer-Verlag, 1994.