Securing Electronic Health Records in the Cloud

David R. Matos [1] Miguel L. Pardal [1] Pedro Adão [1, 2] António Rito Silva [1] Miguel Correia [1]
[1] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal
[2] Instituto de Telecomunicações, Lisboa, Portugal
{david.r.matos,miguel.pardal,pedro.adao,rito.silva,miguel.p.correia}@tecnico.ulisboa.pt

ABSTRACT

Health care institutions gather and store sensitive information from patients with the goal of providing the best care. The medical history of a patient is essential to guarantee that the right diagnosis is achieved and help the clinical staff act in the shortest time possible. This information is highly sensitive and must be kept private for the responsible staff only. At the same time, the medical records should be accessible by any health care institution to ensure that a patient can be attended anywhere. To guarantee data availability, health care institutions rely on data repositories accessible through the internet. This exposes a threat since patient data can be accessed by unauthorized personnel. It is also extremely difficult to manage access to data using standard access control mechanisms due to the vast amount of users, groups and patients and the constant adjustment in privileges that must be done to maintain confidentiality.

This paper proposes a solution to the difficulty that is managing user access control to a complex universe of user data and guarantee confidentiality while using cloud computing services to store medical records.

CCS CONCEPTS

•**Security and privacy** → *Software security engineering;*

KEYWORDS

Privacy, Confidentiality, E-Health, Security Architecture, Information Flow, Isolation, Client Platform Security

ACM Reference format:
David R. Matos [1] Miguel L. Pardal [1] Pedro Adão [1, 2] António Rito Silva [1] Miguel Correia [1]. 2018. Securing Electronic Health Records in the Cloud. In *Proceedings of P2DS - EUROSYS'18, Porto, Portugal, April 2018 (P2DS)*, 6 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Electronic Health records (EHR) [11] are kept and managed by health care institutions. There is no control with the way records are kept and stored and it is extremely difficult to grant access to third parties or different health care institutions. This is a problem since EHRs contain essential information to treat patients, such as clinical history, allergies, blood type, genetic conditions and so on. A patient that wants to receive treatment in a different institution needs to ask his usual doctor to release his medical records and send them to his new doctor. This process is time consuming, complex and, in some cases, may be impossible if the patient is unable to do this (for instance, if he is unconscious).

It is complex to implement protocols that enable health care institutions to share patient's data due to heterogeneous systems, legacy systems, legislation and software limitations. Löhr, et al. [22] proposed a solution that was based on trusted virtual domains so that health care institutions could share access to their application. This solution granted access to patients data but restrict the access to a specific patient, once an institution was granted access to the system it could read every record on it. This freedom of access violates several privacy concerns.

Patient's data (EHR) should be available to any institution, however it is not viable to implement a nation wide web application to serve every health care institution and its patients since each health care institution has its own requirements and budget constraints. Using cloud service providers to store and manage EHR access can be a solution. This way each health care institution could still use its applications but instead of storing locally patient's data, they would store it and access it using public clouds. Public cloud services are capable of scaling and provide availability since they can be accessed through the Internet. Using public clouds instead of in house data centers can also reduce costs [2].

However, managing access control is a complex and thorough task. RBAC [13] is capable of constraining access to the right groups of users by using well defined roles with privileges. However, proper operation of RBAC requires that roles fall under a single administrative domain or have a consistent definition across multiple domains, making its implementation in such heterogeneous system almost impossible.

This paper proposes a novel approach for the problem of storing EHRs in public clouds. It focuses in ensuring data confidentiality and integrity of the EHRs with a user access control based on the lattice model [10]. This model provides multilateral security without the complexity of managing roles and groups of users from RBAC. It can be easily implemented in a tuple space, as proposed in the architecture of a possible system that implements this approach.

The remaining of this article is structured as follows. Section 2 describes the related work and similar solutions to secure EHRs. Section 3 lists the requirements of a cloud storage service for Health records. Section 4 describes in detail how the lattice model can be applied to the health care data model. Section 5 presents the system architecture and describes each component and how they interact. Section 6 describes the protocol to have access to stored data. Section 7 suggests some techniques to guarantee data anonymity. Section 8, concludes this work with a critical analysis.

P2DS, Porto, Portugal

DOI: 10.475/123_4

2 CONTEXT AND RELATED WORK

Access control includes all the mechanisms that manage access to objects (files to read, write and execute) from entities (users, machines and processes). Almost every operating system implements access control. One of the most used mechanisms is RBAC—Role Based Access Control [13]. In this mechanism access to objects is given to roles, and an entity (user or group of users) can only access an object if it belongs to the corresponding role. For example, only the administrator role can delete records from the database, meaning that a user must belong to the role of administrators in order to delete records.

This access control model can be difficult to manage and maintain in a system where there are too many roles and each role has specific privileges. When a new permission needs to be given it is more safe to create a new role instead of using an existing one, in order to avoid granting access to unauthorized users. This, in turn, will generate several roles that will make the administrators work more tedious. To avoid these problems one can use Access Control Lists (ACL) [28]. In ACL the privileges of an object are stored in a list or vector. ACL is largely used in today operating systems. Windows [14] and Linux [15] are some examples.

Multilevel security [7] is used in systems in which users belong to a well defined and strict hierarchy. In this security mechanism each object has a minimum hierarchy level and subjects can only access an object if they belong to, at least, that hierarchy level. This mechanism was based in the military case where documents have a level (open, confidential, secret and top secret) and can only be accessed by officials with a clearance at least as high as the level of the document. This model has several disadvantages because it is too simplistic. For example, an official with high clearance can access any document, and in some cases only a group of people from a specific department should have access to some documents.

The Multilateral Security [25] model intends to solve the limitations of Multilevel security. In this model objects are secured not only by hierarchy levels (vertically) but also by groups (horizontally). This model reflects the security mechanisms in organizations where employees are distributed in a hierarchy but also belong to a specific department inside the organizations, and so, for example, a Chief Financial Officer that has a high clearance may be restricted to access documents from the Department of Innovation and Development. There are several models that implement Multilateral Security. One of the most used is the Lattice Model [10]. This model uses a combination of hierarchy levels, like Multilevel Security, with code words that represent groups or departments inside an organizations. Each object contains a minimum acceptable hierarchy level and a set of codewords. A user can only access an object if he belongs to a high enough hierarchy level and he has every codeword of that object assigned to him. This model is mainly used by military organizations [10] and is the one used to manage user access in this paper.

Implementing a complex system in the cloud requires an efficient coordination service. The best known service of this kind is probably ZooKeeper [18], but it does not tolerate malicious (Byzantine) replicas, on the contrary DepSpace [5] is a Byzantine Fault-Tolerant Coordination Service built on top of BFT-SMaRt [4]. It provides a secure tuple space implementation and can be deployed in the cloud. DepSpace supports a database used to manage user access which store the lattice model entries.

DepSky [3] is a Byzantine Fault-Tolerant storage service for the cloud of clouds paradigm. It works by using a collection of cloud storage services that are managed remotely by a client library. The users' files are stored encrypted in public clouds. A secret sharing scheme [19, 30] is used to securely store the keys in the cloud providers. It also uses erasure-codes [8, 16, 17] to reduce the required storage for the user's files.

Some solutions to secure health records stored in cloud have been presented. Löhr, et al. [22], presented a solution to guarantee data availability using cloud to support the applications used by health care institutions. The presented solution used Trusted Virtual Domains to secure access to the applications by different users and organizations. Ahuja et. al. [2] presented the state of cloud computing in healthcare. In their study the authors show the advantages in terms of cost and complexity in migrating systems used by healthcare institutions to the public cloud. Li et. al. [21] presented a solution to save Personal Health Records (PHR) in the cloud. PHRs are medical records maintained by patients. Other initiatives were implemented such as: e-health in Austria [29], German health card (eHC) System [33], Taiwan Electronic Medical Record Template (TMT) [26] and in Portugal it was implemented a system to authenticate patients of the National Healthcare Service [1]. All these solutions aim to solve the privacy issues of storing clinical data in the cloud, however, none of them focus in keeping the existing systems while providing a highly available storage service with a flexible access control mechanism. In this paper we propose a new approach that allows medical institutions to keep their systems while using a cloud storage service with an access control mechanism designed for the medical case.

3 REQUIREMENTS

This paper presents a system to store Electronic Health Records in a public cloud. The system must be able to deal with the heterogeneity of all the systems used by health care institutions, guarantee secure storage of patients data, data availability and integrity.

3.1 Users

The users that access health care systems have different roles and privileges. Besides the group of medical staff and patients, other institutions must be granted access to these systems. Such institutions include the government, social security, insurance agencies and medical suppliers. The identified users for the system are divided in seven main groups:

- Medical staff: includes all the doctors, nurses and therapists. They can only access clinical data of their corresponding patients;
- Patients: users who can only access their own information (PHR and EHR);
- Insurance agencies: can only access medical bills in order to pay reimbursements;
- Social security and government entities: can consult taxes, compensations and absence subsidies;
- Administrative users: access to financial and personal information of the patients and medical staff;

- System administrators and operators: have privileges to keep the system running and manage access keys;
- Researcher: for statistics and anonymous patient records.

There are several groups / departments and, in each group, there is a well defined hierarchy. For example, in the group of medical staff there could be a group of nurses and some nurses have more privileges (chief of team) than others (interns). These groups of users motivate the implementation of multilateral security.

3.2 Availability

In terms of availability, patient's records should be accessible from any computer with access to the Internet. This requirement is fundamental because a patient must have the freedom to be attended in any health care institution. The system must be also capable of handling high loads of traffic that can be generated by unpredictable affluence of diseases and possible DDoS attacks. Cloud services are capable of adapting the number of running servers to the current traffic. Also, the use of fault-tolerant systems increases the level of availability.

3.3 Confidentiality

There are confidentiality levels within the records. Some records can be publicly accessed while other should only be accessed by a restricted number of users. The proposed solution assumes four levels of confidentiality:

- Private: includes the patient records (PHR), employees files and private information of the hospital;
- Clinic: includes patient diagnosis and records (EHR), treatments and prescribed drugs;
- Research: data used for research purposes. The records in this level cannot be sufficient to identify patients;
- Public: information that can be accessed by anyone.

Once again, the fact that there are several levels of confidentiality for each group of users motivates the use of multilateral security.

Most records in the system have strong confidentiality levels (patient's records should only be visible by the patient and the responsible medical team). However, once a user accesses a file, even if at some point he no longer have access to the file, he could copy that file to external storage or print it. So there is no way to completely remove the access to a file. In this paper the confidentiality of a file or record is limited to a version. Meaning that, if a file with version number 1 was granted to a user U, then U will always have access to that version of the file. Even if, at some point, his access to the file is removed, he can still consult that version. Only future versions are then restricted. This confidentiality limited to the version of the file is not new and was implemented in other systems [9].

4 THE LATTICE MODEL IN HEALTH CARE

The lattice model is a model that can be used to implement multilateral security. It uses codewords to define the various groups (horizontal security) and describes the relations between the hierarchy levels (vertical security).

The lattice model in this solution uses the four confidentiality levels (private, clinic, research and public) as the hierarchy levels of the users and the seven groups (Medical Staff, Patients, Insurance, Social Security and Government Entities, Administrative Users, System Administrators and Operators, and Researchers) as codewords to give access. Besides the groups of users, there are codewords to identify teams, departments and institutions. Every lattice record has a codeword with an id that identifies the object (can be an EHR or a file in the data repository). Finally, since it does not make sense to remove a file that was accessed by a user (once he as accessed to the file he can copy it to an external storage or print it so there is nothing that can be done to remove that file), the system adopts an append-only approach.

An example of a lattice record can be: (PRIVATE, {medical, "HSM", 123456789}). In this example a user can only access this record if he is in PRIVATE level and has every keyword. In this case he must be in the medical group, be a part of "Hospital Santa Maria" (HSM) and must be given access to the record id "123456789". Given that this is an append only system, when a user is given access to a file he does not gain access to a future version of that file, since it would have a different file id.

The lattice model is sufficient to manage user access control in the system and can be easily implemented in a database with a tuple space data structure. This model is also very flexible since there is no need to create a new group or role every time a new privilege is created. A new codeword is sufficient and assigning codewords to users is simpler than associating users to groups and roles. Another disadvantage of associating users with groups is that it is difficult to know which privileges a user gains when he is assigned to a group, in some cases it is only required a sub set of the privileges of the group, which is impossible to do in RBAC without creating a new group.

5 SYSTEM ARCHITECTURE

Figure 1 shows the system architecture of a secure storage service for health records using cloud providers. The proposed solution does not use a single cloud service provider to store all the records, but instead it uses a cloud of cloud paradigm to increase availability, confidentiality and integrity. The system can be divided into three main groups: Data Repository, Healthcare Institutions Systems and Lattice Data Store.

In the figure health care systems connect with the data repository and the lattice data store. The idea behind this approach is that, instead of implementing a complex system that serves all the institutions, only the databases and security services are upgraded. This approach is not as costly as to implement a full system and does not require the users to learn how to use the new system.

5.1 Data Repository

The data repository is implemented using DepSky [3]. This storage solution was designed to be implemented on top of at least $n = 4$ cloud service providers ($n > 3f + 1$). DepSky is configured in mode CA (confidentiality and availability) meaning that it employs techniques of secret sharing [20] which combines symmetric encryption with a classical secret-sharing scheme. Data is fragmented in blocks using an erasure code in such way that at least $f + 1$ blocks are required to recover the original file and f or less blocks are not enough to give any useful information about the file. The symmetric keys used to encrypt the files are also partitioned through the

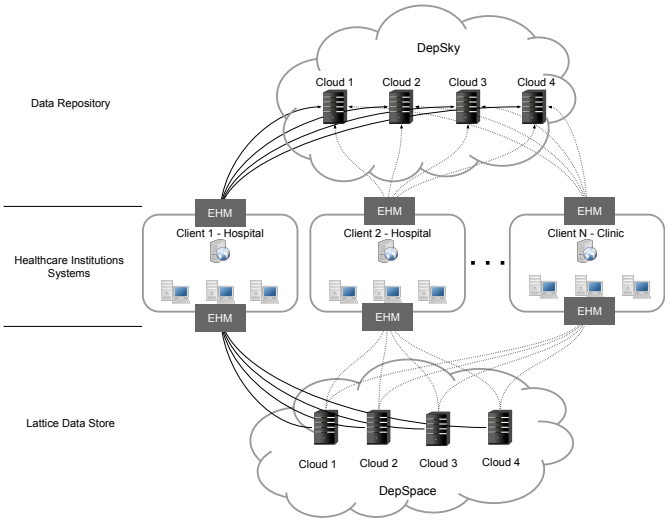

Figure 1: System architecture.

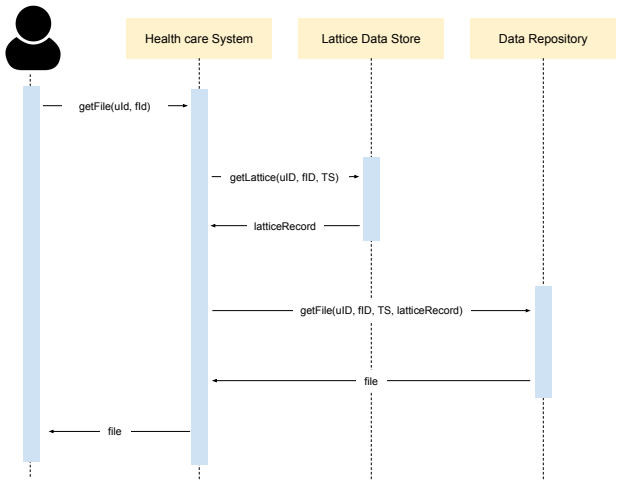

Figure 2: Sequence diagram.

replicas. If a user could access to at most f fragments of the keys he could still not reconstruct the keys with that information.

DepSky is used to store EHR of patients and other information that is not included in the EHR, such as x-rays, EMR files and so on. Using a cloud of clouds to store this sensitive information improves the confidentiality level since an attacker needs to successfully compromise the security of more than f clouds. Even using the lowest possinle n ($n = 4$) an attacker would need to compromise at least 2 clouds so he could read the stored files. If the cloud provider is configured properly, then the probability of an attacker compromising two clouds is very low.

5.2 Healthcare Institutions Systems

The healthcare institutions systems are represented in this architecture because it is assumed that the institutions keep using them. It would be more costly to implement new systems than keeping the current ones running [22]. However, instead of using their own data repositories (SQL databases, file servers, etc.) these systems would use DepSky [3]. This way, for example, a file of a patient who was attended in Hospital A would be accessible by Hospital B because the data repository is accessible through the Internet.

5.3 Lattice Data Store

The lattice data store is a database that saves lattice records. These records are usually in the form of (Hierarchy Level, codeword 1, codeword 2, ..., codeword n). A tuple space is enough to store these records. In this architecture DepSpace [5] implements the data store that saves these records. It was designed to be deployed in $n = 3f + 1$ servers. In this example each server is located in a different cloud. This approach, like DepSky, increases confidentiality and availability, since an attacker would need to successfully compromise more than f servers in order to corrupt the system.

In this architecture there is a separation of concerns. DepSky is responsible for storing data (files and EHR) and DepSpace is responsible for storing access rules (lattice records). This separation of concerns to different systems was inspired by the Certification Authorities [6] in which there is a third party that authenticates users in a system. This third party is a well known and trustworthy service and its only purpose is to ensure that a user has access to a certain object.

5.4 E-Health Middleware

The E-Health middleware (EHM) is the only software component that needs to be developed for the system. It consists of a library that provides an API for the health care system such that they can transparently access the repositories in the cloud. This module is also responsible for communicating with the lattice data store and the data repository, abstracting the complexity underneath. This module is developed as a software library and must provide an interface with exactly the same methods as the old database drivers. Thus, all that each health care system needs to do is to update its database driver to this e-health middleware.

6 AUTHENTICATING USERS

Authenticating a user and checking his privileges is a process that is executed by every component in the system. Figure 2 describes the process of authenticating a user and checking his privileges. At first the user contacts the health care system and asks for a file. The health care system then contacts the lattice data store to retrieve that user lattice record and the lattice record of the file. Once this record is retrieved the e-health secure middleware evaluates if the privileges are valid (i.e. if the user has access to that version of the file) and if he does then the health care system contacts the data repository for the file (giving the lattice record as a key). The data repository then sends the file to the health care system and the user has access to the file. Since the user only contacts the health care system he does not realize the complexity underneath, and thanks to the e-health middleware, the heath care system still uses the same interface as if it was contacting its own database server or file repository.

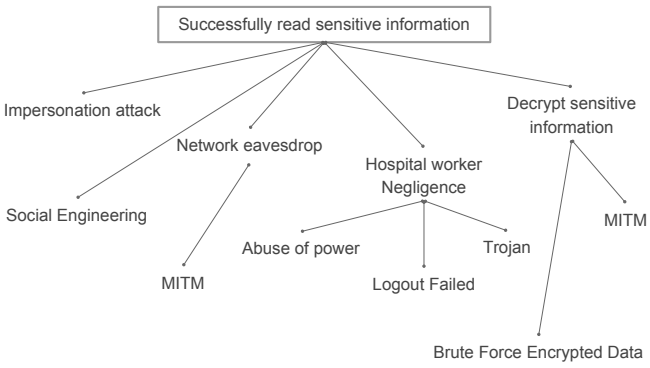

Figure 3: Threat Tree.

7 DATA ANONYMITY

Ensure data anonymity in a database is a challenge. The database with patient records should be accessible to calculate statistics and patterns that are useful for research purposes and help in decision making. For example, if a hospital detects an affluence in flu cases then this information can be used to increase the number of employees and order extra medication. However, collecting this information and still guarantee anonymity of the subjects is extremely difficult. There have been presented some solutions, such as Datafly [31], a middleware that substitutes certain fields in patient records in order to guarantee anonymity. This solution seems to solve the problem but it actually fails in some cases. For example, if an attacker who wants to access a record of a certain patient and he already knows specific details like the birthday, address and one medical condition, then he could execute several queries with refine parameters and eventually he would get, as a result of the query, a single row that can only belong to the target patient. Another approach can be to suppress fields but this fails when the attacker execute several queries in order to complete his personal database.

One of the strongest solutions against these attacks is to substitute a subset of the fields of a record with general information. This technique is called k-anonymity [24, 27, 32] and the main idea is to substitute k-fields of the record with general data that does not change the outcome of the query. This method can be implemented in the E-Health Middleware (EHM).

A useful tool to evaluate the security of a system is Threat Trees (or Attack Trees [12]). The main idea behind this tool is to construct a tree whose root is a possible attack or exploit that can occur in the system, and the nodes of the tree identify possible causes of the attack (vulnerabilities and exploits). This model can serve both as an evaluation tool and as a tool to find countermeasures for the attack.

Figure 3 represents threat tree for the proposed system. Although there are several attacks that can occur, from DDoS to record forgery, this tree focuses on the attack that reads unauthorized data. This attack was chosen because the main goal of this project is to guarantee patients privacy and the whole system was designed with this goal. Although the system provides high level of availability and integrity offered by DepSky and DepSpace protocols, it could still be possible for an attacker to read sensitive data.

In the figure, an attacker can successfully read unauthorized data if he manages to impersonate another user using his access credentials, use techniques of social engineering, listen to network traffic, take advantage of an employee mistake, or decrypt sensitive information. From this set of possible attack two are technical and the remaining three take advantage of user negligence. According to the protocols that support the system, the technical attacks can be avoided if the system is configured correctly (using strong keys to encrypt data, reliable communication protocols such as TLS and regularly renovate encryption keys). The attacks that exploit incorrect behavior from the user are more difficult to avoid. These attacks are normally the most successful. Humans tend to err more than machines [23]. Several techniques can be adopted to minimize the probability of a social engineering attack [34]. Inviting a team to perform such an attack can provide valuable information. Most of the attacks can also be avoided if the users of the system have a correct training and follow strong policies.

8 CONCLUSION

Managing user access control is an extremely difficult and continuous work. Keeping track of the users that should not have access to data and provide access to new users is a delicate task but is essential to guarantee confidentiality of patient's data. The solution presented in this paper solves the problem of user access control and proposes an architecture that decreases the risk of eavesdrop by using a cloud of clouds approach. The presented solution takes into account the complexity of systems used by health care institutions and thus the proposed architecture, which is based on a cloud service with a simple interface, facilitates the migration of data to the new repository and is compatible with almost every web application framework. It is also worth mentioning the impact in terms of performance of such implementation, according to the cloud storage service [3] experiments, the performance overhead is minimal.

Acknowledgements This work was supported by the European Commission through project H2020-653884 (SafeCloud) and by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013 (INESC-ID).

REFERENCES

[1] Projecto Pegasus. http://www.cartaodecidadao.pt. (????). Accessed: 2016-01-30.
[2] Sanjay P Ahuja, Sindhu Mani, and Jesus Zambrano. 2012. A survey of the state of cloud computing in healthcare. *Network and Communication Technologies* 1, 2 (2012), 12.
[3] Alysson Bessani, Miguel Correia, Bruno Quaresma, Fernando André, and Paulo Sousa. 2013. DepSky: dependable and secure storage in a cloud-of-clouds. *ACM Transactions on Storage (TOS)* 9, 4 (2013), 12.
[4] Alysson Bessani, João Sousa, and Eduardo EP Alchieri. 2014. State machine replication for the masses with BFT-SMaRt. In *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on*. IEEE, 355–362.
[5] Alysson Neves Bessani, Eduardo Pelison Alchieri, Miguel Correia, and Joni Silva Fraga. 2008. DepSpace: a Byzantine fault-tolerant coordination service. In *ACM SIGOPS Operating Systems Review*, Vol. 42. ACM, 163–176.
[6] Matt Blaze, Joan Feigenbaum, and Jack Lacy. 1996. Decentralized trust management. In *Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on*. IEEE, 164–173.

[7] Air Force Studies Board. 1983. Committee on Multilevel Data Management Security, Multilevel Data Management Security. *National Academy Press* 1 (1983), 983.

[8] Christian Cachin and Stefano Tessaro. 2006. Optimal resilience for erasure-coded Byzantine distributed storage. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*. 115–124.

[9] Byung-Gon Chun, Petros Maniatis, Scott Shenker, and John Kubiatowicz. 2007. Attested append-only memory: Making adversaries stick to their word. In *ACM SIGOPS Operating Systems Review*, Vol. 41. ACM, 189–204.

[10] Dorothy E Denning. 1976. A lattice model of secure information flow. *Commun. ACM* 19, 5 (1976), 236–243.

[11] Gary Dickinson, Linda Fischetti, and Sam Heard. 2004. HL7 EHR System Functional Model Draft Standard for Trial Use. *Health Level* 7 (2004).

[12] E.Amoroso. 1994. *Fundamentals of Computer Security Technology*. Prentice Hall.

[13] David Ferraiolo, Janet Cugini, and D Richard Kuhn. 1995. Role-based access control (RBAC): Features and motivations. In *Proceedings of 11th annual computer security application conference*. 241–48.

[14] Sudhakar Govindavajhala and Andrew W Appel. 2006. Windows access control demystified. *Princeton university* (2006).

[15] Andreas Grünbacher. 2003. POSIX Access Control Lists on Linux.. In *USENIX Annual Technical Conference, FREENIX Track*. 259–272.

[16] R. Halalai, P. Felber, A. M. Kermarrec, and F. Taani. 2017. Agar: A Caching System for Erasure-Coded Data. In *Proceedings of the IEEE 37th International Conference on Distributed Computing Systems*. 23–33. DOI: http://dx.doi.org/10.1109/ICDCS.2017.97

[17] James Hendricks, Gregory R. Ganger, and Michael K. Reiter. 2007. Low-overhead Byzantine fault-tolerant storage. In *Proceedings of 21st ACM SIGOPS Symposium on Operating Systems Principles*. 73–86.

[18] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed. 2010. ZooKeeper: wait-free coordination for Internet-scale systems. In *USENIX Annual Technical Conference*.

[19] H. Krawczyk. 1993. Secret sharing made short. *Proceedings of the 13th International Cryptology Conference – CRYPTOfi93* (1993), 136–146.

[20] Hugo Krawczyk. 1993. Secret sharing made short. In *Advances in CryptologyfiCRYPTOfi93*. Springer, 136–146.

[21] Ming Li, Shucheng Yu, Kui Ren, and Wenjing Lou. 2010. Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings. In *Security and Privacy in Communication Networks*. Springer, 89–106.

[22] Hans Löhr, Ahmad-Reza Sadeghi, and Marcel Winandy. 2010. Securing the e-health cloud. In *Proceedings of the 1st ACM International Health Informatics Symposium*. ACM, 220–229.

[23] Mr Ian Mann. 2012. *Hacking the human: social engineering techniques and security countermeasures*. Gower Publishing, Ltd.

[24] Adam Meyerson and Ryan Williams. 2004. On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 223–228.

[25] Kai Rannenberg. 2001. Multilateral security a concept and examples for balanced security. In *Proceedings of the 2000 workshop on New security paradigms*. ACM, 151–162.

[26] Hsiao-Hsien Rau, Chien-Yeh Hsu, Yen-Liang Lee, Wei Chen, and Wen-Shan Jian. 2010. Developing electronic health records in Taiwan. *IT Professional Magazine* 12, 2 (2010), 17.

[27] Pierangela Samarati and Latanya Sweeney. 1998. Generalizing data to provide anonymity when disclosing information. In *PODS*, Vol. 98. 188.

[28] Ravi S Sandhu and Pierangela Samarati. 1994. Access control: principle and practice. *Communications Magazine, IEEE* 32, 9 (1994), 40–48.

[29] Thomas Schabetsberger, Elske Ammenwerth, Stefan Andreatta, Gordon Gratl, Reinhold Haux, Georg Lechleitner, Klaus Schindelwig, Christian Stark, Raimund Vogl, Immanuel Wilhelmy, and others. 2006. From a paper-based transmission of discharge summaries to electronic communication in health care regions. *International journal of medical informatics* 75, 3 (2006), 209–215.

[30] B. Schoenmakers. 1999. A simple publicly verifiable secret sharing scheme and its application to electronic voting. *Proceedings of the 19th International Cryptology Conference* (1999), 148–164.

[31] Latanya Sweeney. 1997. Guaranteeing anonymity when sharing medical data, the Datafly System.. In *Proceedings of the AMIA Annual Fall Symposium*. American Medical Informatics Association, 51.

[32] Latanya Sweeney. 2002. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 05 (2002), 571–588.

[33] Thilo Weichert. 2004. Die elektronische Gesundheitskarte. *Datenschutz und Datensicherheit* 28, 7 (2004), 391–403.

[34] Ira S Winkler and Brian Dealy. 1995. Information Security Technology? Don't Rely on It. A Case Study in Social Engineering.. In *USENIX Security*.