TECHNISCHE UNIVERSITÄT WIEN

VIENNA UNIVERSITY OF TECHNOLOGY

DISSERTATION

PIPE: Pseudonymization of Information for Privacy in e-Health

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Doktors der Technischen Wissenschaften

unter der Leitung von

Univ. Prof. Dipl.-Ing. Dr.techn. A Min Tjoa
Institut für Softwaretechnik und Interaktive Systeme (IFS), 188
und
Univ. Prof. DDr. Gerald Quirchmayr
Institut für Distributed and Multimedia Systems, Universität Wien
und
Univ. Ass. Dipl.-Ing. Dr.techn. Mag.rer.soc.oec. Edgar Weippl
Institut für Softwaretechnik und Interaktive Systeme (IFS), 188

eingereicht an der Technischen Universität Wien

von

Dipl.-Ing. Mag.rer.soc.oec. Mag.rer.soc.oec. Bernhard Riedl, Bakk.rer.soc.oec.
Mat.Nr.: 0103517
Färbereiweg 33, 3860 Heidenreichstein

Wien, 08.09.2008

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Ich versichere, dass ich diese Dissertation bisher weder im In- oder Ausland in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

Wien, 08.09.2008

"[. . .] All that may come to my knowledge in the exercise of my profession or in daily commerce with men, which ought not to be spread abroad, I will keep secret and will never reveal. [. . .]"

Hippocratic Oath, Hippocrates 4th century BC

"Security is not a product but a process. Its more that designing strong cryptography into a system; it's designing the entire system such that all security measures, including cryptography, work together."

Bruce Schneier, Communications of the ACM, 1999

Contents

Kurzfassung:

Heutzutage umfassen die vorrangigen Herausforderungen im Gesundheitswesen die Senkung der Kosten für medizinische Dienstleistungen bei gleichzeitiger Steigerung der Behandlungsqualität der Patienten. Eine Möglichkeit dieses Ziel zu erreichen ist die Implementierung eines Elektronischen Gesundheitsakt (ELGA) Systems, welches auch das Ausführen von medizinischen Standard-Prozessen ermöglicht. Da diese landesweiten medizinischen Archive ein vielversprechendes Ziel für mögliche Angreifer darstellen, ist die Bevölkerung besorgt um die Wahrung ihrer Privatsphäre. Diese Befürchtungen und der Mangel an existierenden Ansätzen erzeugt den Bedarf ein System zu entwickeln, welches über ausreichende Sicherheit verfügt. Dieses System soll sowohl die Privatsphäre der Benutzer garantieren als auch den Zugang zu den Gesundheitsdaten unter die strikte Kontrolle der Patienten stellen.

In dieser Dissertation diskutieren wir unser System PIPE (Pseudonymization of Information for Privacy in e-Health). PIPE unterscheidet sich von bisherigen Lösungen dahingehend, dass es für den sicheren Einsatz in der primären als auch in der sekundären Nutzung von Gesundheitsdaten geeignet ist. Zuerst untersuchen wir verschiedene Methoden, Prinzipien und Techniken in den Gebieten Sicherheit und Schutz der Privatsphäre. Danach erarbeiten wir die Eckpfeiler eines sicheren ELGA Systems. Abschließend zeigen wir, wie PIPE die Mängel existierender Ansätze löst.

PIPE kann sowohl als Basis zur Implementierung einer sicheren ELGA Architektur, als auch zur Erweiterung eines bestehenden Systems genutzt werden.

Stichwörter: Privatsphäre, Sicherheit, e-Health, Pseudonymisierung, Autorisierung durch Verschlüsselung

Abstract:

Today, the health care sector is driven by the need to reduce costs while simultaneously increasing the service quality for patients. One major aspect to reach this goal is the implementation of an EHR (Electronic Health Record) system which also supports the execution of medical standard processes. Nevertheless, these nation-wide medical storages are a promising goal for attackers. Thus, people are naturally concerned about their privacy. These concerns and the lack of existing approaches to provide a sufficient level of security raise the need for a system that guarantees data privacy and keeps the access to health data under strict control of the patient.

In this thesis we discuss our approach PIPE (Pseudonymization of Information for Privacy in e-Health), which differs from existing approaches in its ability to securely integrate primary and secondary usage of health data. First of all, we elaborate on existing methods, principles and techniques in the fields of security and privacy. Afterwards, we work out necessary cornerstones of secure EHR systems. Finally, we show how PIPE provides solutions to shortcomings of existing approaches.

PIPE may be used as a basis for implementing secure EHR architectures or as an extension to existing systems.

Keywords: Privacy, Security, e-Health, Pseudonymization, Authorization by Encryption

1 Introduction

In 1948, in its constitution, the World Health Organization (WHO) defined health as "a state of complete physical, mental and social well-being and not merely the absence of disease or infirmity" [155]. Half a decade later, this statement has been enlarged to include the ability to lead a "socially and economically productive life" [157]. The WHO also strengthened the term health promotion under which they sub sum "the process of enabling people to increase control over, and to improve, their health" [156].

To support these demands, decisions of people themselves, medical staff of governmental as well as non-governmental institutions need to be focused on the humans' well-being instead of only dealing with illnesses and injuries. Policy makers in all industries amongst the different sectors have to be aware of the consequences for peoples' overall health, because a working society is based on healthy people. The health sector should not only be responsible for providing clinical services, but see health promotion as an essential part of their work.

The WHO further argues, that people have to be educated throughout their lives about chronic illness and injuries. They propose an holistic thinking about people's health status [91]. Preventive medicine can only result in a good overall health status of the individuals, if carried out regularly. The results have to be compared with the patients' medical history to examine unusual changes. Hence, preventive medicine relies on sufficient documentation and statistic data for further comparison [134]. Latter demand raises the need for a statistical data repository and researchers who develop and work with the health care system's metrics. For example, the Austrian Cancer Register[1)] publishes average rates for certain types of cancer in different regions.

1.1 Background

Due to the enormous quantity of data produced in the health care sector, handling peoples' medical information is still cost-intensive and slow. Though, in the last decade, the IT-sector developed storage systems with almost appropriate capacity for health data. On the one hand, prices for setting-up, maintaining as well as information storing and handling services, are declining [55], on the other hand, communication tasks are still slow and expensive [5].

The historical reasons of missing the ability to store, handle and transport data outside of health care providers systems like hospitals servers or general practitioners' computers lead to the development of a diversity of legacy systems which are used in the e-health sector. Therefore, one major goal of today's e-health services is to harmonize standard processes and data-exchange

[1)] http://www.statistik.at/web_de/statistiken/gesundheit/krebserkrankungen/index.html

formats, because the availability of sound information is essential for health care providers' decisions regarding the patients' care and thus on the quality of treatment and patients' health [74].

The idea of a nation-wide electronic health record (EHR) has been introduced within the past several years as a method for improving communication and collaboration between health care providers. The EHR would increase the efficiency of handling medical data and therefore reduces the costs, because the EHR promises massive savings by digitizing medical data like diagnostic tests and images [100]. Recent numbers published by the OECD point out, that the share of the national spendings for operating the health care system, referred to the gross domestic product (GDP) went up from an average of 7.8% in 1997 to 8.5% in 2002. In the previous period from 1992 - 1997, the percentage has been almost unchanged.

The non-profit research organization Rand Corporation conducted a study on adopting the EHR in the US under the assumption that 90% of the health care providers would use it and found out that using the EHR could result in more than $81 billion saving per anno in the US [41]. Another estimate comes from the Center for Information Technology Leadership [140]. They proposed a cost reduction of $77.8 billion in the US, which is equal to 5%. In literature, these numbers range from 5% to 13% [12].

More than two-thirds of all general practitioners work in small and medium based practices [30]. Regarding these form of health care providers, Wang et al. measured an estimated net benefit of $86,400 over a five-year period for every health care provider after inventing the EHR. The participants mainly profit from savings in drug expenditures, avoidance of double-taken medical images and reduced errors in administrative tasks. Nevertheless, the net difference differed between a loss of $2,300 and a benefit of $330,900 for the study subjects. Wang et al. concluded, that several key factors like the kind of patients treated or the kind of health care provider may influence the results [151]. Miller et al. regarded an increase in quality and a cost reduction of approximately $20,000 a year for every health care provider [80].

As life-long medical information about patients, for example, allergic reactions to drugs is available with such systems, the EHR would also help reducing the alarming number of more than 98,000 cases of death a year in the US [67], caused by adverse drug events (ADE). Additionally, the costs for ADE, which counts up to $175 billion a year in the US [40, 41], could be lowered because the health care teams [100] are provided with additional information, for example guidelines for drug interactions or sophisticated data produced by decision support systems. Therefore, decision support systems on drug interaction are needed [13, 39, 74, 145] to assure a better quality of patients' treatment.

Even if the first patient-centered medical record has been setup in 1907 in the Mayo Clinic [148], it is yet still difficult to take all stakeholders' needs into consideration. Information which has to be processed (semi-)automatically has to match a certain pattern and may pass several stages of editing and viewing. As long as the medical system lacks standard processes, which reflect the stakeholders' needs, the EHR cannot be used efficiently, because the information flow and its controls need to be standardized. Nevertheless, the EHR could be the opportunity to develop and implement processes in hospitals and practices. This would lead to a medical world, which software architects are able to map to a workflow management and execution system based on the underlying hardware infrastructure [54]. Further, the implementation of standard processes within the EHR and accordant workflows, may also help to increase the medical service quality [70, 78].

Besides the organizational aspects of implementing the EHR, patients are concerned about their privacy [116]. For example, a history about substance abuse, abortion or HIV infection could result in discrimination or harassment. Different stakeholders like insurance companies or employers demand the disclosure of anamnesis, diagnosis and treatment data for billing or hiring. Equipped with this information, insurance companies could use the sensitive medical data to deny health coverage or to increase insurance premium for affected persons, whereas employers might negate employment.

Following Givens, in 1995 4 out of 5 patients were concerned about their privacy. Almost 25% have already been a victim of a privacy invasion. Slightly less than 60% declined to hand over personal information if they thought it was too personal or not necessary for a certain kind of business. This number is especially interesting, because just 40% made the same decision five years earlier. Alan Westin concluded, that these concerns are triggered by "distrust in institutions and fear of technology abuse" [52].

In 2005 the California Health Care Foundation (CHCF) carried out their "National Consumer Health Privacy Survey". 11,000 People were asked about their opinion regarding [24] their privacy. More than two-thirds declared, that they are "somewhat" or "very concerned" about their privacy. Still, nearly 60% would participate in an EHR system if they gain access to better coordinated medical treatment. In contrast, one out of eight patients would deny storage of their data in an EHR system, even if they expose their health to risk. Another interesting fact was that slightly more than 50% are willing to share their data for secondary usage in anonymized databases in order to advance the healthcare system, if the data is processed with privacy protection.

Adams et al. conducted a pilot study with 80,000 households (225,000 residents). Five health care providers' settings were linked into an EHR system for 6 months. The whole project started

in September 2000 and ended in March 2003. In this project a lot of effort was put into the confidentiality concept. For example, the Hampshire electronic health record pilot project team provided a help desk together with information on leaflets as well as a web page. As a direct result, only 10 people opted out to have their data processed by the new system. The help desk has been used 82 times. [2]

Most governments of developed countries and their data protection institutions recognized the concerns of their citizens and setup laws and regulations towards their citizens' privacy. Since 2005 the processing and movement of personal data is EU-wide regulated by law through the Directive 95/46/EC [43]. Further a citizen's right of privacy is recognized in the Article 8 of the European Convention for the Protection of Human Rights and Fundamental Freedoms [34]. Subsequently, in the EU several domestic acts have been installed, for example the Austrian Data Protection Act [106] dictates strict regulations on the processing of personal data.

The Article 29 Working Party, which is the data protection institution in the European Union, pointed out several constraints, for example that every patient should be in full control of their data. Claims like that are necessary to ensure the patients' trust in the EHR [44]. The majority of the Article 29 Working Party's demands are already reflected in European law, but some very essential points are still in discussion.

In 2006 the United States Department of Health & Human Service Health issued the Insurance Portability and Accountability Act (HIPAA) which demands the protection of patients data that is shared from its original source of collection. The HIPAA also issued the Protected Health Information (PHI) statement, in which they define health information as "any information, whether oral or recorded in any form or medium" that (i) "was created or received by a health care provider, health plan, public health authority, employer, life insurer, school or university, or health care clearinghouse" and (ii) "relates to the past, present, or future physical or mental health or condition of an individual; the provision of health care to an individual; or the past, present, or future payment for the provision of health care to an individual." [14].

On the one hand the US Department of Health wants to maximize availability and linking of different health care providers data repositories, on the other hand they applied the privacy rule of "limiting uses and disclosure to the minimum necessary". In detail, the HIPPA allows the disclosure of medical information by a health care provider for treatment [147]. This disclosure does not explicitly involve the patient's authorization or the patient's request. On the contrary, what amounts to the minimum is left up to particular health care provider, not the patient [102]. This makes the US privacy regulations significantly weaker than the European ones.

1.2 Motivation

> "The risks are high: even a few privacy violations could lead to user distrust and abandonment of context-aware systems and to lost opportunities for great enhancements."
>
> *Jiang and Landay [61]*

However, although national laws demand the protection of health data stating privacy as a fundamental right of every citizen, in practice existing approaches and EHR applications have security shortcomings which also directly influences the patients' privacy. One technical solution towards a secure EHR architecture, would be to encrypt the medical information and setup a key infrastructure. Unfortunately, as medical data tends to be very large and encryption is a highly time-consuming operation, encrypting all data would not be feasible.

The usage of other privacy enhancing technologies (PET), like pseudonymization seems to be a promising approach in order to guarantee patients' privacy, because it assures efficient confidentiality and integrity as well as high availability of stored information. Nevertheless, existing pseudonymization approaches have drawbacks that pose a major threat to the privacy and confidentiality of stored patient data [107, 109, 110, 112–114].

In this thesis we introduce a detailed description of a new system for the pseudonymization of health data that differs from existing approaches in its ability to securely integrate primary and secondary usage of health data and thus provides a solution to security shortcomings of existing approaches. We further provide novel concepts for data sharing, authorization and data recovery that allow recovery of the access to the health care records if the security token for access to the medical records carrying the keys (e.g. a smart card) is lost or stolen.

Compared to existing approaches, our concept PIPE (Pseudonymization of Information for Privacy in e-Health) does not depend on a patient list, which reflects the association between the patient's identification and medical data or a breakable algorithm. Instead, we base our architecture on a layered security structure that guarantees that the patient is in full control of her data. The concept can be used as an extension to EHR applications but also as basis for national EHR initiatives.

Based on this discussion, we introduce the occurring *Research Questions* in the next section.

1.3 Research Questions and Goals of this Thesis

As aforementioned, if it comes to the introduction of an EHR system, decision makers have to bear several vital factors in mind. Experts expect an intense change in medical teams' working environment as well as in patients' daily life [60]. In addition, citizens are concerned about their privacy [24, 52].

In the following we outline the *Research Questions* of this thesis.

Research Question A Regarding pseudonymization as the chosen privacy-enhancing technology: what are the cornerstones and principles of a secure pseudonymization approach?

Research Question B How can a secure and efficient measurement for safeguarding the users' keys be provided? What are the occurring costs of such a vital add-on?

Research Question C What are the workflows of a secure EHR system which is based on pseudonymization? Which administrative and operational workflows besides those, which are necessary to realize the secure backup system requested in *Research Question C*, are needed?

Research Question D Besides the default access to the medical data, how can ad-hoc access in case of a medical emergency be provided?

Based on these questions we define the goals of this thesis. The expected results lead to the establishment of a rule-set for secure pseudonymization. These introduced principles will be the input for the design of PIPE's architecture. Our focus in this architecture lies on enhancing the security while also providing a sufficient level of efficiency and usability. Moreover, we will introduce a secure backup keystore for the users' keys.

The main objective of this thesis is to establish a workflow framework for e-Health applications by applying the architecture discussed above. We will provide a generic solution by abstracting the methods of this novel framework. Thus, the gained contribution can either be used as basis for new EHR systems or as security enhancement of existing e-Health applications.

Another requirement for an EHR architecture is to provide the functionality of an emergency access to medical data. In contrast to daily usage, a case of emergency lacks the time to authorize

the health care providers to access a individual data set of a patient. Thus, we define as another goal of this thesis to permit a reliable ad-hoc access technique for emergency doctors.

1.4 Methodology

In this section we outline the methodology which we apply to answer the research questions.

- First of all we conduct a secondary literature study and investigate the status quo of existing privacy-related technologies. Then we discuss the necessary basics to assure appropriate security in storage systems. In parallel we conduct workshops with the members our business partners to discuss the gained results.

- In the second step we compare different existing EHR systems regarding (i) their type of storage, (ii) how information is processed, (iii) which backup approach for lost keys is applied and (iv) how emergency access is realized.

- Afterwards, we abstract and combine the advantages of the discussed systems to define the necessary security constraints for the establishment of a secure e-Health system based on pseudonymization.

- With the achieved results we are able to setup a secure architecture inheriting an authorization model and a backup keystore based on the design research approach.

- To assure a sufficient level of security we conduct probability-based analyzes and investigate the created backup approach. As costs are also important for future EHR implementations based on PIPE, we introduce a set of equations to measure the economic factors of our backup keystore. Then we discuss a real-world example to evaluate the occurring cost structure.

- After presenting the security, efficiency and usability constraints of our system, we are consequently able to realize the main outcoming of this thesis, a formal representation of the necessary information flows in a secure pseudonymized EHR system. The goal is to provide the necessary administrative and operative workflows. In other words, these workflows cover the user and data management as well as authorization functions. Moreover, we introduce a method for emergency access to defined subsets of the patients' anamnesis, diagnosis and treatment data.

- Finally, we implement a proof-of-concept prototype as a feasibility demonstration check. To ensure the usability of our approach, we conduct a simulation in a real-world environment

and take the users' acceptance as feedback to improve our approach.

Based on the chosen methodology we introduce the structure of this thesis in the next section.

1.5 Structure of the Thesis

The thesis is structured as follows. First of all we introduce basic concepts on which we based our deliberations. Thus, we determine the topic privacy with the focus on privacy-enhancing technologies in Section 2. Afterwards, we elaborate on the necessary security basics and special needs for security in e-Health in Section 3. Moreover, we investigate the security and privacy constraints in existing systems. In Section 3.9 we conclude how a secure pseudonymization approach can be realized, and therefore answer to *Research Question A*.

In Section 4 we introduce the cornerstones of our approach PIPE and further present the security stack and authorizations mechanisms. We discuss *Research Question B*, the establishment of a backup keystore in Section 4.5. As questioned in *Research Question C* we state the workflows in a formal way in Section 5 and show our proof-of-concept prototype in Section 6. The explicit answer to *Research Question D* is given in Section 5.4. Finally, we present our conclusions in Section 7 and review the Research Questions in Section 7.1.

2 Privacy

> "No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks."
>
> *United Nations, Universal Declaration of Human Rights, Article 12 [146]*

> "The right to one's person may be said to be a right of complete immunity: to be let alone."
>
> *Cooley [33]*

> "data processing systems are designed to serve man; [...] they must, whatever the nationality or residence of natural persons, respect their fundamental rights and freedoms, in particular the right to privacy, and contribute to economic and social progress, trade expansion and the well-being of individuals"
>
> *European Union, Directive 95/46/EC [43]*

Not only the security, but also the privacy of information is a cornerstone of a modern society. Nevertheless, nowadays the rights of an individual's privacy is often invaded by governmental organizations or companies in order to conduct surveillance operations or to create a buyer's profile.

In 2000, the Wall Street Journal conducted a survey about the fears of Americans for the upcoming century. They found out that 29 % ranked the concern about "erosion of personal privacy" at the first or second place. No other issue scored more than 23 % [137].

Following the US, several laws have been installed in the European Union and subsequently in its members states. In 2002, the EU adopted a framework decision to "fight more efficiently against terrorism" [42]. Nevertheless, in the EU several institutions still try to prevent legal acts comparable to the patriot act in the EU. The "Center for European Policy Studies" demands a uniform standard for data protection in all member countries which participate in international data-exchange systems. Moreover, they advocate for the importance of a personal privacy intrusion decrease [6].

The Article 29 Working Party was setup under the Directive 95/46/EC of the European Union as a data protection unit. Based on European Law, they demand the following principles: [44]

1. Limitation Principle: This principle relates to the Article 6(1)(b) of the directive and prohibits data processing beyond the purpose of collection.

2. Data Quality Principle: Following Article 6(1)(c) data has to be accurate and up-to-date. Moreover, neither is it allowed to collect more data than necessary, nor has unnecessary data to be stored.

3. Retention Principle: It is not allowed to collect data further they purpose of task-related processing.

4. Information Requirements: As demanded in Article 10, the data subjects have to be informed about collection and processing of their data.

5. Data Subject's Right of Access: For accuracy checks and to keep their data up-to-date, the data subjects are permitted (Article 12) to access any of their stored personal information.

6. Security Related Obligations: Due to Article 17, data controllers have to implement appropriate technical and organizational measures for information protection against accidental or unlawful destruction of unauthorized disclosure.

Besides the legal situation in the US and the EU, another acceptance factor for the electronic health record (EHR) will be the peoples' trust in such a system. This goal can be achieved by the assurance of patients' privacy in combination with appropriate security measurements and the communication of the means used in understandable language.

In the following, we define the term privacy in Section 2.1 and discuss several techniques to implement the aforementioned demands in Section 2.2.

2.1 Definition

> "Privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others."
>
> *Westin [153]*

We define privacy as the assurance that only the data owner herself and persons authorized by the data owner are allowed to conduct reading or writing operations on the data owner's personal data. Not only sensitive information itself, but also the association between a certain individual

and her data claims to be protected for privacy reasons. We sub sum all violations against both of the latter principles under the term privacy invasion.

> "[...] privacy is a perception which differs from person to person, changes over times and emerges from a society's communication practices."
>
> *Wahlstrom and Quirchmayr [150]*

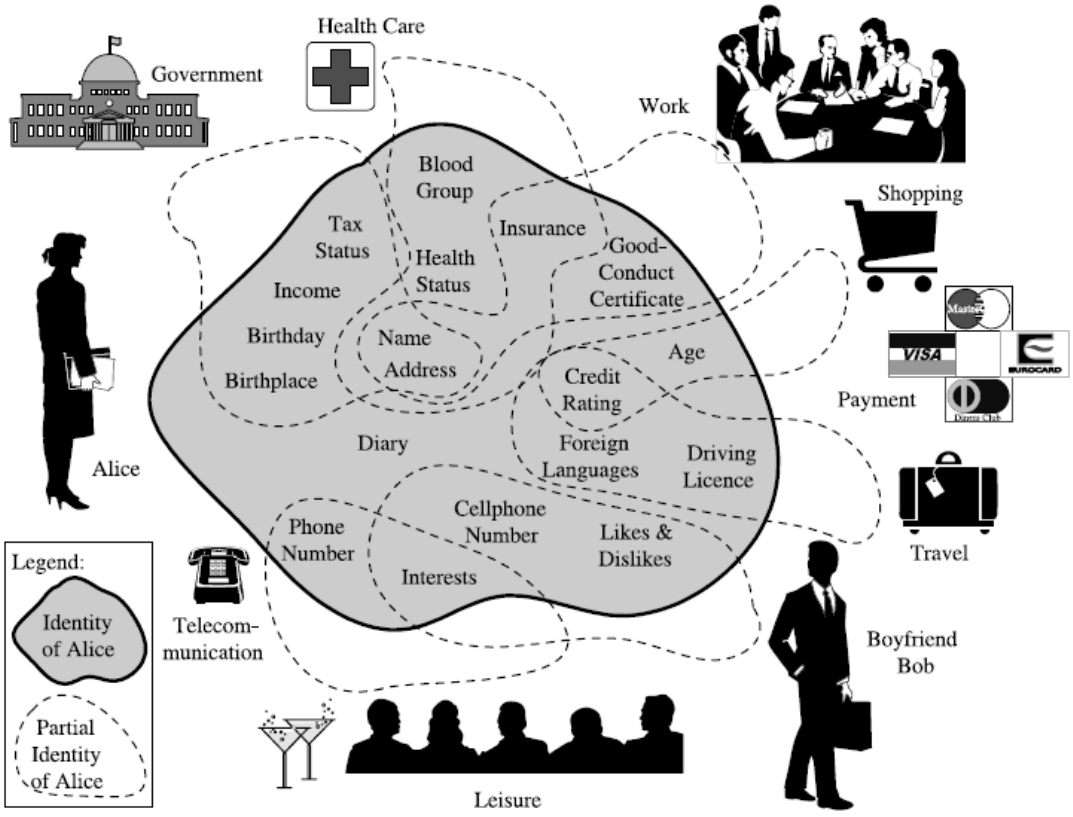

Figure 1: Partial Identities of Alice [31]

Persons's opinions about the desired privacy differ. Identity management provides an overview of the personal data about the individual. This discipline shows the complete and all partial identities of a certain person and her interactions with her environment (cf. Figure 1). Different stakeholders, like for example an insurance company or the person's employer, need insights on these partial identities [31]. Though every citizen still has the fundamental right of privacy. This means, that she is the only person who decides, based on her perception, which parts of her complete or partial identities she want to share, because she is the owner on the data.

Thus, she has to be put in charge of her personal information as far as possible. Based on this fact, four basic principles emerge, on which privacy is merely based on: [25]

1. To ensure the trust in an application, users have to be informed about what data is stored and how it is processed [28].

2. It is on the behalf of each user which data she wants to make accessible to which persons. In other words, every user should be in full control of her data.

3. The amount of information's personal-identifying attributes has to be reduced in order to seek anonymity. This means, that whenever it is not necessary to store the association between a user and her data, it should be left out.

4. From a privacy point of view, personal data should never be collected only for the purpose of storing. There has to be necessity to complete a certain task.

Privacy can only be granted by implementing a security policy, because privacy is based on the security attributes of confidentiality and integrity [75]. In charge for the design, implementation, deployment and enforcement is the Chief Privacy Officer (CPO). The person who fulfills this role has to be involved in any Enterprise Resource Planning (ERP) system development or maintenance task [25].

As an ERP system reflects the companies' requirements and business continuity, not only the security should be main focus during planning and modeling, but also the assurance of privacy and user' trust has to be weaved into the development process. This task, which is also controlled by the CPO may be based on standards, best-practices like the aforementioned principles and extensive reviews of documents and source-code artifacts. Moreover, the CPO has to issue a privacy-related document including all actions undertaken on personal data. For example, the W3C standard P3P (Platform for Privacy Preferences)[2)] emerged during the recent years for the deployment of a privacy policy on the Internet.

Moreover, Agrawal et al. stated that privacy metadata has be to recorded with all medical-related information. This list of security-related constraints, does not only contain the purpose of usage, but also (i) all persons who are allowed to access the document, (ii) the retention period, which defines how long the data is stored and (iii) a group of users to whom the data can be given out [3].

[2)] `http://www.w3.org/P3P/`

2.2 Privacy-enhancing Technologies

A PAT (privacy-aware technology) is a privacy add-on for an already existing system. The actions undertaken for granting privacy may include the usage of passwords, encryption, tailored access controls or secure communication behavior. These trust-creating features have been left out during the design and implementation of certain systems, though they can still be appended later with increased effort. Nevertheless, it is important to ensure a security and privacy aware development process after upgrading an architecture without the focus of privacy to a PAT. [25]

In contrast to a PAT a PET (privacy-enhancing technology) is a solution which is mainly focused on achieving the maximum of possible privacy in a system [15, 22, 25, 27, 130, 138]. The creation of a PET includes the deployment of a privacy-aware security policy, which not only covers technical, but also organizational aspects [150]. Furthermore, the CPO has to set-up a privacy assuring process for development, execution, maintenance and evolution of the system [25].

There are different possibilities and levels for PETs which are related with the purpose of personal data usage. We introduce a comparison of different PETs and their constraints in the forthcoming sections.

2.2.1 Anonymization

The basis of every ERP system is a database. This storage on the one hand holds the data and on the other hand allows queries against the stored information based on an access model. The database is structured into one or more tables, depending on the business needs.

In a simplified medical database, two tables exist. One holds personal data of the patients and another one medical data, in other words the anamnesis, diagnosis as well as treatment data. Every table-row is associated with a primary key, which unambiguously identifies each dataset. The relation between a particular patient and her data is created by the storage of the primary key of the patient's table as the secondary key in the medical data. This technique, called database normalization helps avoiding the storage of patients' personal data which each medical dataset and reduce redundancy [32].

In an architecture without the focus on privacy, any intruder who gets access to the database, may steal the stored data. As no preventive action against a confidentiality, integrity or privacy breach has been undertaken, the attacker may work with the stolen data immediately. If the company operating the system had for example encrypted the data with a single administrative key, and therefore created a simple PAT solution, the attacker has to gain at least a copy of that specific key as well to compromise the data.

The opposite of such an architecture aforementioned is a system which is based on anonymization [97, 104, 143]. Total anonymization means removing the secondary key of the patients table in the medical data table and deleting the patients table. Hence, no personal information is stored in an anonymized system [97, 104, 143]. The purpose of this kind of systems are for example probability theoretical evaluations or anonymized polling. In other words, the person's identity does not matter [99].

Besides the secondary key of the patient's table additional personal information might exist within the medical data table. The process of identifying and separating personal from the related medical data is called depersonalization [104]. After depersonalization two different data repositories exist. One which holds the personal information and a second one where all the residual data is stored. This separation is the basis for a successful anonymization by lowering the risk of privacy invasion [107, 109, 110, 112–114].

Depersonalization may help to filter out any personal-related information, but the risk of re-identification of a certain individual in a anonymized database also merely depends on factors like the quantity and distribution of data and its classes. For example, only one patient might have a rare disease. If the statisticians need the political district for their research and this information is also available in the anonymized database, they might already guess who this patient could be. Nevertheless, these cases are highly unlikely and need to be observed on data entry. Another possibility to ensure privacy would be encryption, but this a time-consuming operation, and not always feasible for large amounts of data, like medical data [107, 109, 110, 112–114].

Levels of Anonymity The appearance of users in the system is another aspect of anonymity. Following Flinn et al., five different levels of anonymity exist [49], which pose the different possibilities for user interaction in a system.

Level 0 No User's Identification: An example would be the access to a public web page without logging functionality.

Level 1 Anonymous Identification: The user authenticates as anonymous user probably with a password. This technique is wide spread for FTP servers. Logs may be stored to identify the user by the network address. Nevertheless, network addresses may be dynamic or the user might connect through a proxy server. For example the web-service Anonymizer[3)] provides different proxy-servers to browse the Internet anonymously.

Level 2 Pen-Name Identification: A user-avatar/nickname together with a password is used for

[3)] `http://www.anonymizer.com/`

authentication purposes. Logs may be stored to identify suspicious behavior, but the user cannot be related to a certain individual. Moreover, users have the freedom to use several pseudonyms and thus alter her usage history fraudulently. Nowadays, level 2 is used for most web 2.0 services like Flickr[4] or WordPress[5]. If a user applies anonymizer service, which prevents the establishment of an usage profile, it is called unobservability.

Level 3 Latent (Potential) identification: The user is known to the system as a person. Every user holds a set of pseudonyms which might be used for communication purposes. This anonymity level may be favored by system operators, because the legal situation in certain countries might demand linking-back of user activities if laws are broken.

Level 4 Usual Identification: The user name and password has been issued by a system administrator. The user may be unambiguously identified by her user name. As she is known as a person to the system and other users, all undertaken actions may directly be associated with this individual. This level is typically used for multi-user network environments.

Level 5 Super-Identification: Stronger authentication methods, like signatures or biometric authentication are used in level 5. This level not only assures that a user may be related with all her actions, but also adds the security constraint of non-repudiation. Access to sensitive information, for example medical data, has to be secured by these means of authentication.

Complexity of Anonymity Another aspect of anonymity is its complexity. It defines the minimum number of entities necessary to unveil the identity of a system's user [51, 129]. The number of people, who have to act together, to reverse the anonymity of a system not only influences a possible privacy invasion, but also may rise or lower the costs and processing time.

Basically, three different possibilities for the so-called Order-N anonymity (OA) exist: [51, 129]

$OA(0)$ Nowadays, an anonymity complexity of 0 is very common for systems with a single administrator. The person inheriting this role may re-establish the identity of a anonymized dataset. This variant is, besides a backup for the person, rather cheap and effective to execute.

$OA(1)$ To avoid easily compromising of the system, $OA(1) - OA(n-1)$ assure that not a single user alone is able to reverse a certain individuals' anonymity. Instead the separation of

[4] `http://www.flickr.com/`
[5] `http://wordpress.org/`

privilege design principle is applied, which can be, for example, realized by the implementation of a threshold-scheme.

$OA(2)$ It is called limited anonymity, if all participating users have to act together to unveil a particular individual's identity. Absolute privacy can only be assured if the user herself has to agree as well.

k-Anonymity As we discussed at the beginning of this section, total anonymization means the removal of all identifying attributes from datasets. Nevertheless, the resulting privacy is gained by a concurrent loss of information. Sweeney outlined that with the appliance of total anonymization the quality of the data is decreased, because some identifying attributes are necessary to conduct research [71,121,135,136]. For example the age or the sex of persons in a medical study is a vital information which may directly influence the findings.

In case anonymized data also holds parts of the personal-related data so-called joining attacks may occur. This form of attack is based on the combination of the attributes of one or more databases [135,136]. In a simplified way we can think of a certain purpose for every database. One database could for example hold persons' medical data, while another one is used as voter list. If an attacker is able to combine the identifying attributes of both databases, she is not only able to re-identify a certain person, but also gain additional information from both databases. Hence, it is important to know what attributes are stored in which database, because a successful joining attack between two databases or more databases unveils all of the underlying partial identities of a certain person. Thus, even if a database 'looks anonymized', the risk of an attack is increased with every public database and identifying attribute. It is also hard to distinguish which data could be used in future attacks [135, 136].

Quasi-identifiers QI are combinations of attributes that can be used to conduct a joining attack with the usage of one or more external databases [71]. For the following example we combine all tables of a database to a single table T. Every tuple of this constructed table T holds QI_{T_i} quasi-identifiers and j non-personal related attributes. Moreover, every person is only depicted by one tuple. We examine the data stored in QI_{T_i} of T by measuring the minimal appearance of multiple entries. This is value is called k-Anonymity [71, 135, 136].

In our example (cf. Table 1), T consists of the quasi-identifiers ZIP-code, age and sex. All combinations of minimal one of these or maximal all three attributes are possible quasi-identifiers. In addition T also holds the diagnosis as an example for a non-identifying attribute. In our case the k-Anonymity for the combination of all three attributes is two, because a minimum of two tuples exist, which hold the same data. If we would only publish the table with the ZIP-code as

ZIP-code	*Age*	*Sex*	*Diagnosis*
1234	27	f	Migraine
1234	27	f	Fever
4567	28	m	Flu
4567	28	m	Hypertension
4567	28	m	Migraine
1234	40	f	Heart Attack
1234	40	f	Aids
4567	65	m	Fever
4567	65	m	Hypertension

Table 1: Example for k-Anonymity where $k = 2$, $QI_T = \{ZIP - code, Age\ and\ Sex\}$

only quasi-identifier the k-Anonymity is increased to $k = 4$.

Regarding the anonymization of databases, it is necessary to balance the k-Anonymity and the quality of the released data. In other words, an increase in the k-Anonymity may go hand in hand with a decrease of the possible usage of the provided information. Nevertheless, it is important to assure k-Anonymity to protect the individual's privacy.

Though, in case of the existence of multiple public databases, simple attacks are still possible, even if k-anonymity is given. For example an attacker could combine two databases, if the identifying attributes are only removed, but the original sort order is kept. To avoid this type of attack, anonymized tables shall be randomly ordered before handing over to the research institution [135, 136].

In case an anonymized data repository has been published as follow-up to an already existing study material another type of attack may occur, if different quasi-identifiers are used. For example, if the age has been left out in the first study, it must not be included in the second one, because otherwise a joining attack could be possible. Hence, it is important to assure, that the identifying attributes of follow-up studies do not hold more information than previous ones. Moreover, the same number of tuples has to be used in both studies, because otherwise attackers may be able to conclude the identity of certain persons by investigating the appended tuples [135, 136].

Anonymization, and its result anonymity, on the one hand assures privacy but also hampers using of data beyond the usage for statistics or aggregations. In the following we discuss pseudonymization, which is a technique for reversible anonymity.

2.2.2 Pseudonymization

Another approach for granting privacy is called pseudonymization. In Greek a pseudonym is a false name, used for example by authors which do not want to share their identity. In the field of information engineering it means a technique where identification data is transformed into a specifier and then afterwards replaced by this specifier. The security of pseudonymization is based on the fact, that this specifier can not be associated with the identification data without knowing a certain secret [97, 112, 114, 114, 139].

In the aforementioned example, the secondary key in the medical data table, which is the primary key of the patients table will be replaced by a pseudonym. The pseudonym itself is stored securely by means of cryptography. Thus, pseudonymization is a form of reversible anonymity. Regarding the complexity of anonymity for pseudonymization, OA is equal the number of persons necessary to unveil the certain individual's identity.

Algorithms for calculating the pseudonym can be based on encryption or hash techniques [73]. If the latter is applied, the only way of assuring reversibility is to store a list where all pseudonyms are kept [18, 48, 98, 99]. The usage of a list is a weak point in the architecture of existing systems for pseudonymization, because if an attacker gains access to this list she is able to establish a link between the identification data and the medical data of a specific patient [112–114]. Encryption provides a more secure alternative for issuing pseudonyms. Instead of storing the pseudonyms, the keys can be held in a secured keystore.

There are several possibilities to calculate a pseudonym. In the most simplest way, pseudonyms are random numbers, but they could also inherit some semantic, too. More sophisticated methods exist which are related to the problem of finding random numbers [21]. Moreover, the place where the pseudonyms are issued, may differ as well. The pseudonyms could be calculated by a central service, or directly on a user's smart card [125].

Unlinkability Guaranteeing privacy with pseudonymization does not only depend on secure pseudonym generation, but is also influenced by the unlinkability. Following our example aforementioned, two tables exist. One which holds the patient's personal data and the secured pseudonyms. The second table consists of the pseudonyms, the anamnesis, the diagnosis and the treatment data. Thus, it is only possible to associate a certain medical dataset with a patient by knowing the necessary secret, in our case the pseudonym.

Nevertheless, it would still be possible to combine single datasets to a patient's medical history, if the same pseudonym would be used for every medical dataset of a certain patient. This is

called a profiling attack. The attacker, who possesses the data may conduct this type of attack to establish a patient's medical history and use it to guess the patient. If enough data exists, that is a realistic attacking vector.

To avoid such an attack, distinct pseudonyms have to be used for any dataset. Furthermore, the usage of a global unique identifier (GUID), for example, to exchange data between health care providers may also lead to the same result. Hence, these form of identifiers have to be avoided, too [25, 45].

Summary From a usability point of view a system with enhanced privacy results in decreased usability. This can not only lead to latency because of the reaction time, but also cause serious issues. For example, a patient needs to confirm a health care provider's access to her data. If she is unconscious she might not be able to react. Hence, the useful information about an allergic reaction to a certain substance cannot be accessed. That could lead to serious injuries or even the death of the patient. It is not possible to help the patient in an anonymized system, but a pseudonymized system may provide a special emergency authentication technique, which still assures the patient's privacy for normal usage [111].

In the following we provide a summary of the introduced techniques.

- Architecture without the focus on privacy: As a privacy-enhancement is not applied, thus a direct link between an individual and her data can be established in case of an attack. An additional encryption may assure the security and privacy of the users' data. If latter is applied, the system would be called a PAT.

- Anonymization: This approach is the complete opposite compared to the previously mentioned one. The identifier is removed from the data. Thus, this technique is resident against privacy invading attacks until k-Anonymity is given. Nevertheless, this technique can for example only be applied in a medical study, in case the person's identity does not have to be re-established later on.

- Pseudonymization: This PET is a trade-off between a standard architecture without the focus on privacy and total anonymization. Moreover, with this technique it is most of the time possible to circumvent encryption, which is a time-consuming operation. If a reliable access concept exists to control the re-association of a certain patient with her data, pseudonymization is nearly as secure as encryption itself.

Anonymization and pseudonymization are both techniques which assure privacy by removing identifying attributes from the anamnesis, diagnosis and treatment data. Hence, it is a important

pre-condition to determine which database fields might hold content which could lead to a privacy-invasion. Especially in the case of the secondary usage of medical data all privacy constraints need to be strictly controlled by the CPO. Several techniques and metrics exist which can be applied to assure that the patient's privacy is guaranteed.

As the topic of privacy protection is evolving fast, the Privacy and Data Security Institute of the Technical University in Dresden provides a terminology paper on their web page[6)], which is updated regularly.

In this section we discussed the term privacy and the techniques anonymization and pseudonymization. In the upcoming section we introduce the cornerstones of security which are the basis to assure the patients' privacy. Then we elaborate on different solutions towards a secure EHR implementation and investigate their security and privacy constraints. Finally, we work out ten principles which need to be followed to develop a secure pseudonymization solution.

[6)] http://dud.inf.tu-dresden.de/Anon_Terminology.shtml

3 Security

Following the WHO proposals, every citizen should be able to lead a socially and economically productive life. Health care systems should not only focus on healing patients after they got certain disease, but also concentrate on preventive medicine and individuals' life-long training. This may only be achieved with sufficient documentation and statistic metrics.

The electronic health record (EHR) is a service which stores or links life-long medical data of every patient. Besides that, the EHR provides a decision support system and holds a drug interaction database. Moreover, the data gained can be used by a social or health insurance company for billing purposes or to handle drug placement by pharmacists.

As medical information within the EHR is sensitive information and needs therefore to be protected against unauthorized access, we focus on the privacy issues that come with this novelty. We setup PIPE to be conform with European law, which dictate strict regulations on the storage and movement of medical data. Our architecture is also able to handle medical information for secondary usage for research purposes by reversible anonymization.

Chhanabhai [29] found out in their survey, that from a health customers point of view, a general concern about the security, privacy and confidentiality of their medical records exists. Due to historical experience or distrust in technology, the main concern is unauthorized access to medical information. Unauthorized access is expected from outside attacks conducted by hackers as well as from inside attacks by employees. Nevertheless, the difference between the people who think, that a traditional paper-based record is more secure than an electronic one, is small [29].

It is therefore necessary to build up peoples' trust in a new EHR system. A strong security concept is the basis to provide confidentiality and privacy of the patients' data. As medical data, especially medical images tend to be very large, encryption is most of the time not feasible. Instead, other privacy-enhancing techniques (PET) have been developed to protect sensitive information [130].

In this section we explain the related security terms. Moreover, we introduce concepts on which a secure EHR system should be merely based on.

3.1 Definition

Traditionally, there are three different fundamental attributes of security: Confidentiality, Integrity and Availability (CIA). Following Avizienis et al. [9] security as well as dependability define the requirements of a reliable system (cf. Figure 2). In their opinion every system may

fail, but can still be regarded reliable, if the frequency of failures is acceptable.

Moreover only authorized actions should be served by a trusted system [28]. Generally, other important metrics for a dependable system is the reliability, which is the continuity of correct service. Safety is the absence of dangerous consequences on the system or its users. Finally, maintainability stands for the possibility for persons with administrative roles to adopt malicious behavior or adopt the system to changed or enhanced business needs [9].

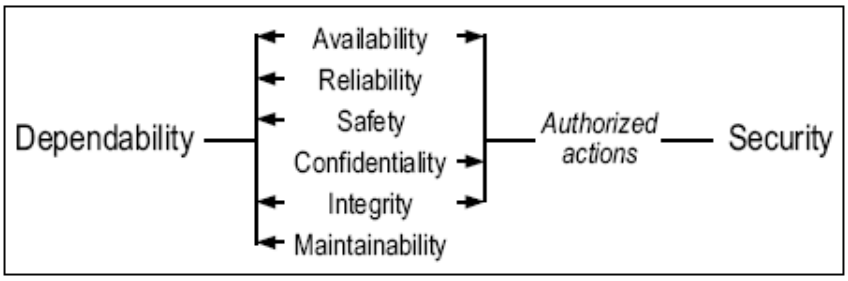

Figure 2: Overview of Dependability and Security Attributes [9]

Organizational or administration security is highly relevant even though people tend to neglect it in favor of fancy technical solutions. The most appropriate security measurements can be bypassed for instance by a successful social engineering attack on a user inside the system, who tells an attacker the necessary passwords [77, 144].

Whether a system is "secure" or not, merely depends on the definition of the requirements. As nothing can ever be absolutely secure, the definition of an appropriate security policy based on the requirements is the first essential step to implement security. A security policy is the rule set which defines the security constraints under which a certain system is allowed to operate and interact. In other words, all restrictions and permissions for persons, objects, information-flows and data storage are defined there [152].

3.2 Systematic Categorization of Requirements

All requirements that we perceive can be traced back to one of the three major security requirements, confidentiality, integrity and availability. A forth requirement, non-repudiation, can be seen as a special case of integrity and availability (i.e. the integrity of message which was sent from A to B).

Confidentiality The perhaps most well known security requirement is confidentiality. It means that users may obtain access only to those objects for which they have received authorization,

and will not get access to information they must not see. The security policies guaranteeing confidentiality are implemented by means of access control.

Integrity The integrity of data and programs is just as important as confidentiality but in daily life it is frequently neglected. Integrity means that only authorized people are permitted to modify data (or programs). The security policy guaranteeing integrity is implemented by means of access control as like above.

Availability It is through the Internet that many users have become aware that availability is one of the major security requirements for computer systems. Productivity decreases dramatically if network based applications are not available or only limitedly available.

There are no effective mechanisms for the prevention of denial-of-service, which is the opposite of availability. However, through permanent monitoring of applications and network connections it can be recognized when a denial-of-service occurs. At this point one can either switch to a backup system, like additional Internet connections from another Internet Service Provider, or take other appropriate measures.

Non-repudiation The fourth important security requirement is that users are not able to plausibly deny having carried out operations. Let us assume that a teacher deletes her students' exam results. In this case, it will be possible to trace back who deleted documents and the tracing records must be so reliable that one can believe them. Auditing is the mechanism used to implement this requirement.

Non-repudiation is one of the major principles in cyber crime investigations [26]. It is a necessity to ensure that any action undertaken can be associated with the individual who conducted it. This demand leads to the definition of digital evidence and it is obvious that this can only be assured if non-repudiation is based on the principles of confidentiality and integrity.

In the next sections we elaborate on the mechanisms that are used to implement the aforementioned requirements (confidentiality, integrity, availability and non-repudiation).

3.3 Cryptographic Primitives

In the following we outline the different cryptographic means, which we use to establish PIPE's architecture (cf. Section 4) and consequently realize the workflow frameset (cf. Section 5). For

further reading about cryptographic and other security techniques we recommend [53, 128].

3.3.1 Encryption

> "Cryptographic techniques have been providing secrecy of message content for thousands of years."
>
> *Kahn [64]*

Cryptography has a long tradition. Humans have probably encrypted and decrypted communication contents since the early days. For example, the so called Caesar encryption is a classical method by which Caesar is said to have used to send messages to his generals [118]. The Caesar's code is a so-called symmetric key algorithm, where both communication partners, named Alice and Bob share the same secret key K.

Nowadays, the Advanced Encryption Standard (AES) is the National Institute of Standards and Technology (NIST) standard for symmetric encryption [85, 115]. Nevertheless, with the usage of symmetric encryption algorithms a previous key-exchange is necessary for secure communication.

In 1976 Diffie and Hellman solved that problem by the invention of the concept of so-called asymmetric key algorithms [36]. In contrast to symmetric algorithms, where the sender and the receiver have to know the same secret, in asymmetric cryptography every participant holds a secret (private) key and a public key, which is disclosed for communication purposes. The latter is also applicable for digital signatures.

A communication between the sender Alice and the receiver Bob is initiated by exchange of the receivers public key. Then, Alice encrypts the message with the public key of Bob. Upon receipt, only Bob is able to decrypt the message by applying his private key to the decryption algorithm.

Currently, two broadly used asymmetric encryption algorithm variants exist. In 1978 Rivest, Shamir and Adleman invented RSA [117], which still is the best known and most used asymmetric algorithm. The security of RSA relies on the problem of factoring large numbers.

In 1985 the use of elliptic curves in cryptography (ECC) was published independently by Koblitz [66] and Miller [81]. In contrast to RSA, the security of ECC is based on the elliptic curve discrete logarithm problem (ECDLP), which is the problem of finding k given points kG and G on an elliptic curve [57].

Compared to RSA, the necessary key length to establish confidentiality with ECC is significantly lower [50, 132], as depicted in Table 2.

ECC (order of base point P)	*RSA (length of modulus n)*
119	512
144	768
163	1024
222	2048

Table 2: Comparison of Key Length for ECC and RSA [50, 132]

In practice, Han et al. measured that an ECC decryption with a key length of 163 bits, which is form a security point of view equally to a 1024 bits RSA key, is five times faster than the RSA pendant. Moreover, the resource usages of ECC is smaller than RSA. Nevertheless, encryption operations are slower [56]. Hence, it depends on the type of application if RSA or ECC should be used.

There are two reasons to establish hybrid techniques, which are a combination of an symmetric and an asymmetric cipher algorithm. Firstly, symmetric algorithms have a higher efficiency and secondly, the asymmetric alternatives use less keys and ease key exchange.

Equation 1 states a comparison of the necessary keys for symmetric cryptographic operations whereas n is the number of the communication partners. Compared to that exponential growing number of necessary keys, for asymmetric encryption only two keys are necessary for every participant.

$$\frac{n(n-1)}{2} \tag{1}$$

Due to the mentioned facts, it is reasonable to combine these two types of algorithms in a way, that the asymmetric variant is used to envelop a symmetric key for secure transport over a distrusted channel like the Internet.

1. Alice requests Bob's public key, either from a certificate authority (CA) or Bob himself. In addition to confidentiality, the usage of a CA assures the identity of a certain person.

2. Alice generates a symmetric session key and encrypts this key with Bob's public key.

3. Alice uses this symmetric session key to encrypt her message and transfers both ciphertexts to Bob.

4. Upon receipt Bob decrypts the enveloped symmetric key with his private key and subsequently uses the gained plaintext key to decrypt the message itself. For further communi-

cation purposes, this symmetric key can be applied.

The TLS (Transport Layer Security) protocol, which is the successor of SSL (Secure Sockets Layer), is another well-known hybrid cryptographic protocol [8]. In contrast to the standalone-software PGP[7)] which also uses hybrid encryption, TLS acts as an additional layer for machine to machine communication. It can either be applied to secure the HTTP (Hypertext Transfer Protocol) protocol and thus form the HTTPS (Protocol) protocol to encrypt for example credit card data or work directly on top of the TPC/IP protocol stack (Transmission Control Protocol / Internet Protocol). The latter is also called the Internet protocol suite, which is the set of communications protocols that implement the protocol stack on which the Internet and most commercial networks run.

As aforementioned, a symmetric algorithm is more efficient than an asymmetric alternative. Apart from that only asymmetric alternatives allow two users, which do not want or are not able to share the same key, to communicate securely. In PIPE we apply asymmetric encryption for example to allow the centralized logic module to communicate with the users (cf. Section 5). Moreover, we use a form of hybrid encryption in our security hull model (cf. Section 4).

3.3.2 Hash Techniques

Regarding Equation 2, a hash function h takes a variable-size input (of any length $*$) $x \in \{0,1\}^*$ and returns a fixed-size (fixed length n) string $y \in \{0,1\}^n$. As this hash-value y, which is sometimes also called a message digest, is a depiction of the original text, a cryptographic hash algorithm can be used to increase efficiency. A hash function is irreversible an can therefore only be used if the original text does not need to be re-established. [87]

In practice hash functions are applied for example on e-mail messages before signing them as we discuss in Section 3.3.3.

$$h : \{0,1\}^* \rightarrow \{0,1\}^n, h(x) = y \tag{2}$$

As stated by Menezes et al. [79] the security of hash functions relies on the following general requirements:

- Pre-image Resistance: given a secure hash function $h(x) = y$ it is impossible to find x for the given hash value y.

[7)] http://www.pgpi.org/

- Second Pre-image Resistance: it is computationally impossible to find two different values x_1, x_2 such that $h(x_1) = h(x_2)$.

- Collision Resistance: it is computationally infeasible to find a pair x_1 and x_2 such that $h(x_1)=h(x_2)$. Due to the possibility of a birthday attack, a hash function output must be at least twice as large as what is required for preimage-resistance.

After Rivest invented the MD5 (Message Digest Algorithm 5) algorithm in 1992, the National Security Agency (NSA) published their SHA (Secure Hash Algorithm) hash function in 1993. Soon after, weaknesses have been found in both algorithms. As no other successor has been released for MD5, the NSA slightly adopted their work and named their new version SHA-1. Thus, nowadays SHA-1 is commonly used in protocols like TLS or SSH (Secure Shell) or applications like PGP [128]. In 2002, the SHA family was extended to include in total five algorithms, SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512.

Similar to the selection of AES [115], the NIST (National Institute of Standards and Technology) announced a contest in 2007 for the next standard of hash techniques, SHA-3 [86].

In contrast to the SHA hash family, HMAC (keyed-Hash Message Authentication Code) is a keyed-hash function [87]. It requires two parameters. One parameter is the input value whereas the second parameter is a secret key K shared between two communication parties.

As stated in the position paper [89] the security of HMAC relies on the security of the respective underlying hash function. Given a hash function $\eta = h_k(r)$, an attacker would be able to calculate the output η without knowledge of the secret key by either finding a collision of the underlying hash function or by finding the output of the compression function with a random and secret initial value. Attacks like these are currently not accomplishable even with the use of SHA-2. Furthermore at this time, known collisions of other hash functions do not show any significant implications on the security of the HMAC scheme, which allows the assumption that the HMAC can be considered as secure [87, 89].

In addition to transferring keyed hashes, HMAC can be used to create a chain of hash values. This functionality can be used to generate a replay-attack resident authentication token [63, 111]. We depict the HMAC cryptographic function in Equation 3. This function allows the initialization of the first value $\eta_{U_1} = h_K(r)$ of the transaction pseudonym chain by first applying a random number or random string r as the argument. This random value may be exchanged even in clear text between the communication parties, because only the secret key needs to be concealed.

$$\eta_{U_i} := h_K^i(r) = h_k(h_K^{i-1}(r)), \quad i = 1, 2 \ldots \tag{3}$$

All other subsequent authentication tokens are based on the first hash value. Subsequently, hash value η_{U_i} is the result of an earlier cryptographic calculation that is based on its respective successor hash value $\eta_{U_{i-1}}$.

In Section 5.4 we discuss how HMAC can be applied as authentication technique to avoid replay-attacks.

3.3.3 Digital Signatures

In contrast to encryption techniques which preserve the confidentiality of information, digital signatures are used for integrity purposes. In asymmetric cryptography, signing of messages works is mirrored to the encryption of messages.

As presented in the previous paragraph, Alice firstly applies a hash algorithm to generate a hash value, which represents the message as the original message would. Afterwards Alice uses her private key to sign the message. As the matching public key is known to Bob, he is able to verify the integrity of the message by (i) verifying the signature of the hash value and (ii) recalculating the hash value.

The elliptic curve digital signature algorithm (ECDSA) [62] is the alternative to signing messages with RSA. In Table 3 we provide an overview of the different speeds of ECDSA and RSA for generating and verifying signatures. Processing of digital signatures with a 1024 bits RSA key is faster than its equivalent, from a security point of view, 160 bits ECDSA. If it comes to a 2048 bits RSA variant, ECDSA with 216 bits has nearly double the efficiency of RSA for generating a signature, but is still much slower in verifying.

Algorithm	*Generate Signature*	*Verify Signature*	*Average*
RSA (1024 bits)	25ms	2ms	13.5ms
ECDSA (160 bits)	32ms	33ms	32.5ms
RSA (2048 bits)	120ms	5ms	62.5ms
ECDSA (216 bits)	68ms	70ms	69ms

Table 3: Comparison of Processing Times for ECC and RSA [50]

To assure the integrity and non-repudiation (cf. Section 3) in PIPE we use digital signatures on the information (cf. Section 5.3.3 and 5.3.5) as well as on the information's metadata (cf. Section 5.3.7 and 5.3.8).

3.3.4 Threshold Schemes

As previously stated, confidentiality of information can be achieved by applying symmetric, asymmetric or hybrid encryption techniques. Moreover, the communication partners may apply digital signatures to assure the transferred information's integrity. In 1968 Liu stated another challenge of the field of information management which can also be solved by cryptographic means [72].

In the so-called Liu's problem [72], a group of scientists are collaborating on a project. As they need to keep all the project-relevant data secret, they decide to lock up the generated artifacts in a safe. Furthermore, they state the demand, that this safe can only be opened if a majority of the group members agree. In Liu's example, the group consists of 11 scientists, which means that a minimum of 6 people have to act together to unveil their secret material. Liu defined that all scientists have equal rights to open the safe. In other words, it does not matter which members of the group participate in unveiling the project's secret. Thus, in this real-world example, the safe has to be secured with a minimum of 462 different locks and 252 matching keys, which is clearly impractical.

About a decade later, Shamir abstracted Liu's problem. He stated that a threshold scheme consists of a secret D, which needs to be protected against unauthorized access. Thus, D is divided into n shares, whereas the knowledge of a minimum of k shares is necessary to reconstruct the secret, or in other words, to open the safe. To establish a hierarchy within the group of the scientists, Shamir recommends to distribute more than one share to certain participants, depending on their status. Nevertheless, even if someone possesses $k-1$ shares, it is still not possible to gain any information about the secret by herself [131]. The latter is also called a perfect threshold scheme.

A threshold scheme, which is the appliance of a secret sharing scheme, may also work as a security method to realize for example the four-eyes-principle. A famous example for this separation of privilege would be the ignition of nuclear weapons during the Cold War. In Section 3.4.6, we discuss further appliances for the realization of secure systems.

In 1979, Shamir published his approach to realize a perfect threshold scheme. His approach is based on polynomial interpolation and the fact, that a polynomial $y = f(x)$ of degree k is uniquely determined by $k+1$ points. In the following we describe the two phases of Shamir's Secret Sharing (SSS) technique.

- Initialization Phase: The dealer is the trusted instance who generates the shares. Firstly, the dealer chooses a random polynom $f(x) = a_0 + a_1x^1 + a_2x^2 + \ldots + a_{k-1}x^{k-1}$ of degree $k-1$,

whereas $a_0 = D$. Afterwards, she generates the secret shares by calculating $\sigma_i(S) = q(i)$ for $1 \leq i \leq n$ and hands out the n shares together with together with the public index i to the participants.

- Reconstruction Phase: To re-calculate the polynom, any set of k or more shares have to be combined to re-construct the secret D, because the knowledge of k points allows to unambiguously determine the polynomial $f(x)$ by using the Lagrange interpolation.

Regarding the threshold scheme constraints, as aforementioned, Shamir's threshold scheme is perfect because it is not possible to gain more knowledge about the secret D with less than k shares. Thus, this technique can be considered secure because an attacker cannot determine which of the calculated points are part of D [82].

As in SSS the secret is encrypted as coefficient a_0 in the random polynom $q(x)$ and the x-coordinates i of $\sigma_i(S)$ are public knowledge, only the corresponding y-coordinates have to kept secret. Therefore, the sizes of the secret shares are equal to the size of the secret which means that Shamir's secret sharing scheme is also ideal [79].

3.3.5 Excursus: Time without End

A brute-force attack against a cipher-algorithm means simply trying out keys until the matching one is found. This can be done for example by a dictionary attack, which could limit the tries if the user has selected a word instead of a random string or number. Nevertheless, in the worst case, all permutations of allowed letters, numbers or special characters for the particular key or password have to be tried out [37, 128].

In Table 4 we state the key length of asymmetric and symmetric algorithms and the maximum of necessary arithmetic operations. The operations are not related with their occurring effort and are therefore only to been seen relatively [132].

ECC (in bit)	*RSA (in bit)*	*AES (in bit)*	*Arithmetic operations (32 bit)*
119	428	56	$1.7 * 10^{19}$
144	768	69	$1.1 * 10^{23}$
163	1,024	79	$1.3 * 10^{26}$
222	2,048	109	$1.5 * 10^{35}$

Table 4: Key Sizes and Required Arithmetic Operations for Brute-Force Attack [132]

A year has about 31 million seconds. Thus, 1 MIPS year is equal to $3.1 * 10^{13}$ instructions, or the possible operations of a computer with the speed of 1 MIPS. Note: A modern computer

may conduct somewhat 100 MIPS. Nevertheless, calculations in MIPS should be taken as an approximation or to get the dimension to solve, for example, a cryptanalysis problem [132].

To enhance the conceivability, we present a comparison of the numbers for the common RSA public-key modulus lengths using the General Number Field Sieve in MIPS [11] together with figures from our universe and earth biology [38] in Table 5.

Attack	*Duration (in MIPS)*
Break RSA 768 bits	10^8
Break RSA 1024 bits	10^{11}
Break RSA 2048 bits	10^{20}
Event	*Duration (in years)*
Evolve a new species	10^6
Evolve a genus	10^7
Evolve a class	10^8
Evolve a phylum	10^9
Evolve from the primeval slime to Homo Sapiens	10^{10}
Closed Universe	10^{11}
Open Universe Low-mass stars cool off	10^{14}
Planets detached from stars	10^{15}
Stars detached from galaxies	10^{19}
Decay of orbits by gravitational radiation	10^{20}
Decay of black holes by Hawking process	10^{64}
Matter liquid at zero temperature	10^{65}
All matter decays of iron	10^{1000}
Collapse of ordinary matter to black hole	$10^{10^{26}}$
Collapse of stars to neutron stars or black holes	$10^{10^{76}}$

Table 5: Summary of General Number Field Sieve [11] for RSA and Time Scales on Earth/in the Universe [38]

3.4 Security Design Principles

Following Saltzer and Schroeder [120], Bishop stated the eight design principles for secure systems [17], which we elaborate in this section. These rule sets are the basis to work with access models and role definitions.

3.4.1 Principle of Least Privilege

> "Least privilege: Every program and every user of the system should operate using the least set of privileges necessary to complete the job. Primarily, this principle limits the damage that can result from an accident or error. It also reduces the

> number of potential interactions among privileged programs to the minimum for correct operation, so that unintentional, unwanted, or improper uses of privilege are less likely to occur. Thus, if a question arises related to misuse of a privilege, the number of programs that must be audited is minimized. Put another way, if a mechanism can provide 'firewalls,' the principle of least privilege provides a rationale for where to install the firewalls. The military security rule of 'need-to-know' is an example of this principle."
>
> *Saltzer and Schroeder [120]*

This principle demands, that only those privileges that are needed to complete certain tasks, are granted to individual users, user-groups or programs [17].

If it comes to control the access to data objects or groups of objects, firstly a security policy has to be developed. It is used to reflect the minimum needs of users or user-groups. Secondly, this policy has to be verified with mathematical means by formal checks [127]. Finally, an access control model (cf. Section 3.6) is the implementation of the aforementioned security policy. This model relies on the proper execution of the operating system's reference monitor to guarantee controlled access to each data object and thus, comply to the regulations of the policy.

Due to its restrictions of how and when privileges are granted, this principle is sometimes called the 'less is more' principle.

3.4.2 Principle of Fail-Safe Defaults

> "Fail-safe defaults: Base access decisions on permission rather than exclusion. This principle, suggested by E. Glaser in 1965 means that the default situation is lack of access, and the protection scheme identifies conditions under which access is permitted. The alternative, in which mechanisms attempt to identify conditions under which access should be refused, presents the wrong psychological base for secure system design. A conservative design must be based on arguments why objects should be accessible, rather than why they should not. In a large system some objects will be inadequately considered, so a default of lack of permission is safer. A design or implementation mistake in a mechanism that gives explicit permission tends to fail by refusing permission, a safe situation, since it will be quickly detected. On the other hand, a design or implementation mistake in a mechanism that explicitly excludes access tends to fail by allowing access, a failure which may go unnoticed in normal use. This principle applies both to the outward appearance of the protection mechanism and to its underlying implementation."

Saltzer and Schroeder [120]

The security policy of a certain system must not only define the access rights of users and objects. As the principle of fail-safe defaults demands that all rights have to be denied by default, the security policy should include a rule set for granting new, adopted or enhanced rights for users and user-groups. The consequence for people inheriting administrative roles in a certain system is, that they are able to maximize the security at setup time and grant the necessary access rights afterwards [17, 120].

For example, a small company has three different user-groups — management, accounting and marketing. By default, the management is able to read the data produced by accounting and marketing. The other two departments are only able to access their own data. In case the marketing user-group needs access to the numbers of the yearly balance to work out a presentation for the customers, a rule in the security policy may demand that only the management is able to grant further access rights. This complies with the privilege of fail-safe defaults.

Moreover, following the management's rules, the marketing should only get access to the aggregated data, leaving out the details. This is a measurement to assure the privilege of least principle, because all actions executed by the marketing department on the detailed accounting data are considered unsecure and not necessary to complete their task.

As discussed in Section 5 in e-Health applications the patient, who is the data owner, has full access rights on her medical information. Unless she wants to equip additional users for example, health care providers with reading or writing permissions, access is denied to all users by default.

3.4.3 Principle of Economy of Mechanism

> "Economy of mechanism: Keep the design as simple and small as possible. This well-known principle applies to any aspect of a system, but it deserves emphasis for protection mechanisms for this reason: design and implementation errors that result in unwanted access paths will not be noticed during normal use (since normal use usually does not include attempts to exercise improper access paths). As a result, techniques such as line-by-line inspection of software and physical examination of hardware that implements protection mechanisms are necessary. For such techniques to be successful, a small and simple design is essential."

Saltzer and Schroeder [120]

Dating back to the 14th century, Occam's razor (aka Ockham's razor) which is attributed to the English logician, as well as the Franciscan friar William of Ockham, was published. This principle states that the explanation of any phenomenon should make as few assumptions as possible. In other words, if all other things are equal, the simplest solution is the best.[8)]

If we apply Occam's razor to today's security engineering, a simple design and implementation is not only desirable to decrease the costs for implementing and maintaining a software system, it also simplifies testing and formal checking. Furthermore, enhancements are also cheaper and it is less likely to run into security vulnerabilities [17, 120].

Nowadays, software is built up of several modules which interact to meet the business needs. These modules can be implemented and tested separately. The principle of economy of mechanism demands to use modules because smaller units are easier and cheaper to implement and test, though the usage of as few modules as possible limits the integration effort. In this case, security is a balancing act between long-time planning and feasibility.

A security pattern, named, single access point, which is directly deduced from this principle, strengthens the need for as few access paths to a certain system as possible [160]. For example, if a company's representatives have to place new orders online, they should preferably all have to pass the same security access paths. A VPN (Virtual Private Network) is one possibility to assure the confidentiality of these orders as well. If this technique is applied there should be no other entry to the company's network or subsequently the company's software.

Another area of application for the security design principle of economy of mechanism is the interaction between several systems. Instead of sharing a homogeneous data- or codebase, the interaction between different software systems is nowadays rather service-based. This means that each system provides a communication interface for reading and writing of data. It is clear, that the used interfaces, which are in fact additional access paths, need to be secured.

In the section Architecture (cf. Section 6.3) we discuss the implementation of our service-based prototype and show how we minimize the necessary access paths in combination with a centralized logic module.

3.4.4 Principle of Complete Mediation

> "Complete mediation: Every access to every object must be checked for authority. This principle, when systematically applied, is the primary underpinning of the

[8)] data from wikipedia, `http://en.wikipedia.org/wiki/Occam's_Razor`

> protection system. It forces a system-wide view of access control, which in addition to normal operation includes initialization, recovery, shutdown, and maintenance. It implies that a foolproof method of identifying the source of every request must be devised. It also requires that proposals to gain performance by remembering the result of an authority check be examined skeptically. If a change in authority occurs, such remembered results must be systematically updated."
>
> *Saltzer and Schroeder [120]*

To guarantee appropriate security, it is firstly necessary to implement security checks for each object [17, 120]. In other words, no object has to be left unsecured. As at the design stage, it is hard to predict the direction in which the system will be developed over the next few years, the aggregation of security checks for several objects may be appropriate in the first version, but may also hamper future secure design because of insufficient means of granularity.

Once more, the access to the objects are checked against the access model (cf. Section 3.6) and is based on a working operation system reference monitor. In our architecture, we enlarged the access model to also reflect additional security layers, which we realized by the usage of encryption as authorization mechanism. We discuss this core functionality of our system in Section 4.3.

3.4.5 Principle of Open Design

> "Open design: The design should not be secret. The mechanisms should not depend on the ignorance of potential attackers, but rather on the possession of specific, more easily protected, keys or passwords. This decoupling of protection mechanisms from protection keys permits the mechanisms to be examined by many reviewers without concern that the review may itself compromise the safeguards. In addition, any skeptical user may be allowed to convince himself that the system he is about to use is adequate for his purpose. Finally, it is simply not realistic to attempt to maintain secrecy for any system which receives wide distribution."
>
> *Saltzer and Schroeder [120]*

In 1883 Kerckhoffs stated, that only military security should not rely on the confidentiality of anything which cannot easily be changed [65]. The apportion of his demand among secure software systems is that, for example, cryptographic algorithms should be known to public, whereas the applied keys or passwords have to be secured. This knowledge has also been reflected in the principles for medical-related privacy-focused architectures, which we discuss in Section 3.9.

Recently, the principle of open design was broken in the deployment of the A5 cipher for telecommunication purposes. In 2001, Biryukov et al. published a real time cryptanalysis of the A5/1 algorithm [16]. Nevertheless, due to the enormous effort which would be necessary to exchange running systems and heavily distributes cell-phones with A5/1 ciphers SIM cards, most of the mobile phone companies still encrypt their communication with this cipher.

Some software engineers still argue, that a simple system is more likely to be attacked than a complex one. The security principle of open design includes the definition that complexity does not add security. Regarding, for example, the design of RSA which is publicly known, the security of this algorithm is based on its simplicity and auditability as well as the concealment of the applied private keys.

3.4.6 Principle of Separation of Privilege

> "Separation of privilege: Where feasible, a protection mechanism that requires two keys to unlock it is more robust and flexible than one that allows access to the presenter of only a single key. The relevance of this observation to computer systems was pointed out by R. Needham in 1973. The reason is that, once the mechanism is locked, the two keys can be physically separated and distinct programs, organizations, or individuals made responsible for them. From then on, no single accident, deception, or breach of trust is sufficient to compromise the protected information. This principle is often used in bank safe-deposit boxes. It is also at work in the defense system that fires a nuclear weapon only if two different people both give the correct command. In a computer system, separated keys apply to any situation in which two or more conditions must be met before access should be permitted. For example, systems providing user-extendible protected data types usually depend on separation of privilege for their implementation."
>
> *Saltzer and Schroeder [120]*

Besides the fact that the access of users to certain objects has to be controlled, it is also a necessity of secure system, to avoid handing over to much power to a particular user [17, 120].

The principle of separation of privilege can, for example, be handled by the appliance of threshold schemes [131]. Two or more individuals of a user-group have to act together to unveil a shared secret. This secret can afterwards be used to initiate a critical action. The action undertaken may influence the future behavior of the system.

Separation of privileges can also be applied in a workflow system. Decisions like, for example a high investment, includes the positive vote of a minimum of a defined number of users of a certain group. Only if enough decision-makers voted in favor of the suggestion, the workflow continues and triggers the desired action.

Moreover, this principle may also help to restrict the granting or denying of access rights for other user. Hence, if not enough system administrators want to change a particular user's access to a certain object, the change is not conducted. Such an implementation of the four-eye-principle also minimizes fraud caused by insider attacks, as long as not enough insiders are working together to compromise the system. Another example to show the necessity of shared access would be the metric level of anonymity (cf. Section 2.2.1).

As EHR systems hold sensitive medical data about patients, not only the access to that information, but also secure disaster recovery scenarios need to be set up. We introduce the appliance of the Separation of Privilege Principle in PIPE's backup keystore in Section 4.5.

3.4.7 Principle of Least Common Mechanism

> "Least common mechanism: Minimize the amount of mechanism common to more than one user and depended on by all users [101]. Every shared mechanism (especially one involving shared variables) represents a potential information path between users and must be designed with great care to be sure it does not unintentionally compromise security. Further, any mechanism serving all users must be certified to the satisfaction of every user, a job presumably harder than satisfying only one or a few users. For example, given the choice of implementing a new function as a supervisor procedure shared by all users or as a library procedure that can be handled as though it were the user's own, choose the latter course. Then, if one or a few users are not satisfied with the level of certification of the function, they can provide a substitute or not use it at all. Either way, they can avoid being harmed by a mistake in it."
>
> *Saltzer and Schroeder [120]*

Resource sharing may help to reduce overheads and therefore costs, but could also lead to a vulnerability in the system [17, 120]. For example, using the cache on a web browser may reduce used bandwidth, but may also compromise the system's security and privacy if other users have access to the same locally stored information.

A shared network connection could lead to a denial-of-service, if too many users are concurrently connected to the system or an attacker uses the security flaw. To avoid such a situation, a proxy-

server may be installed to block certain request and control the available resources [17]. In our architecture PIPE we use a centralized service which also inherits that functionality (cf. Section 4.2).

Besides the sharing restriction regarding this principle, Saltzer and Schroeder highlighted another demand. If a new business need arises for a certain system, decision-makers always have the choice of making or buying it. A common argument infavour of or against individual software is, for example how appropriate a standard library would be compared to a self implemented software. Moreover, the time-to-market (TTM) and the necessary expenses heavily influence such a decision. When it comes to secure software development, the make-or-buy decision is not only influenced by the factors mentioned, but also by the necessary extensive security testing for the new module and the integration testing time and resources needed.

Most of the time, customizing a COTS (commercial-of-the-shell) product does not meet all the needs, but might be the better solution from a security point of view. Moreover, an active product's community may detect vulnerabilities faster and thus help to close them. For example, nowadays, a lot of software is available licensed under General Public License (GPL). One success factor for such an open source product is a skilled development team and the participation of the community that uses the product.

3.4.8 Principle of Psychological Acceptability

> "Psychological acceptability: It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimized. If he must translate his image of his protection needs into a radically different specification language, he will make errors."
>
> *Saltzer and Schroeder [120]*

The goal of this principle in theory is to make security as transparent as possible. No additional effort should be added, neither the usability decreased [17].

It is common in modern operating systems to deploy a strong password policy. One constraint is the "3 of 4", which means the presence of a minimum of three of the four groups of characters, (i) letters, (ii) capital letters, (iii) special character and (iv) numbers. Moreover, strong passwords require a minimum length to comply to the security policy. As it is hard to remember these types of passwords for users, one could use the following approach to simplify the generation of

strong passwords: Imagine a sentence which includes and/or ends with a special character. Take the first letter of every word and combine it to make a password. Otherwise, users may break the company's security policy by sticking post-its on their screens.

In a system, where the security is hard to understand and learn, users will always try to find an easier and faster way to achieve their goal. Hence, security has to be usable as well, because otherwise people will work against their own protection. Other preferable methods from a users point of view would be a hardware security token like a smart card or the usage of finger-prints recognition.

In our prototype we try to maximize the security's transparency. In addition we propose the usage of smart cards instead of long passwords to enhance the usability (cf. Section 6).

Most of the security design principles can only be applied after an appropriate authentication mechanism. The authenticated user may then conduct her granted actions after being controlled by a access model.

3.5 Authentication

Authentication means proving that a person is the one she claims to be [88]. A simple example illustrates what authentication is about. If a user logs on to the system, she will usually enter a name for identification purposes. The name identifies but does not authenticate the user since any other person can enter the same name as well. To prove her identity beyond all doubt, the user must enter a password that is known exclusively to her. After this proof the user is not just identified but (her identity) also authenticated.

Just as in many other areas, the most widely spread solutions for authentication are not necessarily the most secure ones. Security and simplicity of use frequently conflict each other. One must take into consideration that what is secure in theory may not mean it is secure in practice because it is not user-friendly; thus prompting users to circumvent the mechanisms. As we have already mentioned in the security design principle of psychological acceptability, in theory it is, for example, more secure to use long and frequently changed passwords. Nevertheless, many users will avoid these mechanisms effectively by writing down their passwords and possibly sticking post-its on their computers [152].

A number of approaches for authentication can be distinguished [159]:

- What you know (e.g. password)

- What you do (e.g. signature)
- What you are (e.g. biometric methods such as face identification or fingerprints)
- What you have (e.g. key or identity card)

In systems with sensitive data, it is desirable to combine more of the mentioned approaches (e.g. identity cards and fingerprints).

In addition to the aforementioned authentication approaches the necessity exists to secure the transfered security code and to unambiguously identify both communication partners before transmitting confidential data. A secure channel in an unsecure network can be, for example, established with TLS to assure confidentiality.

Furthermore, all individuals participating in the communication can be identified by the usage of a mutual authentication protocol in combination with a CA or trusted third party (TTP) [95]. The latter has to verify the association between a certain user and her digital certificate or public key, so Alice and Bob can trust each other, even if it is a first-time communication. Thus, the security attributes of confidentiality and integrity are assured.

Following Otway and Rees [95] we present an practical example for a mutual authentication protocol with a mutual trusted third party.

Before authentication can take place, each party has to possess a public/private key pair. The relation between a user and her public key is known to the TTP. Moreover, the particular private keys are only known by their owners.

1. Alice asks the TTP for Bob's public key. Upon receipt she verifies the authenticity of Bob's public key by verifying the signature via the TTP.
2. Afterwards she generates two random numbers RN_1 and RN_2 and selects a mathematical operation M like addition or subtraction. She subsequently encrypts all three attributes with Bob's public key and forwards these ciphertexts to him.
3. Bob uses his private key to encrypt Alices message. He uses the gained plaintext random numbers RN_1 and RN_2 and applies the selected mathematical operation M to calculate the result R'_1.
4. Afterwards, he queries the TTP for Alice's public key and signature. Then, he proves Alice's identity via the TTP. Bob chooses a random number RN_3 and uses Alice's public key to encrypt the result of his mathematical operation R'_1 as well as RN_3.

5. Upon receipt, Alice decrypts Bob's result R_1' of her transmitted operation and separately calculates the result R_1. If both results, R_1 and R_1' match, Bob has been unambiguously identified and authenticated, because the TTP guarantees Bob's identity and as only Bob possess his private key, it was only possible for him to decrypt Alice's message.

6. Alice uses the numbers RN_3 and R_1 to apply the mathematical operation M and calculates the result R_2'. Then, she encrypts the gained result with Bob's public key and sends this ciphertext to him.

7. Bob again uses his private key to decrypt Alice's message. Afterwards, he calculates the result R_2 based on result R_1, the random number RN_3 and the mathematical operation M and compares this number with R_2'. If both numbers match again, Alice has been identified and authenticated as well.

A strong authentication is the basis to establish a reliable access control, which we describe in the next Section.

3.6 Access Control

> "Access control constrains what a user can do directly, as well as what programs executing on behalf of the users are allowed to do. In this way access control seeks to prevent activity that could lead to a breach of security."
>
> *Sandhu and Samarati [124]*

Access control is used to grant access (reading or writing operations) to specific objects (e.g. files) only to those who are authorized.

Access control can only work with a reliable authentication. Only if the user's identity can be established reliably it is possible to check the access rights. Access control can take place on different levels. Everyone who has ever worked with a networked computer will know the rights supported by common operating systems such as read access, write access, and execution rights.

3.6.1 RBAC

> "It is generally accepted that RBAC is more suited to health care than other access control mechanisms to meet the requirements for the security of health care information."

Zhang et al. [161]

With the invention of multi-user systems and server-client technology the need raised to control the access to certain objects on a single machine. Several different access control models were published. Irrespective of the form of access control (DAC [122], RBAC [46, 123], or MAC [69]), each access can be described in terms of a triplet (S, O, Op). S stands for the subject that is about to conduct an operation (Op) on an object (O). A specific mechanism of the operating system (often referred to as reference monitor) then checks whether or not the access is to be permitted. RBAC offers additional benefits compared with other access controls, like DAC and MAC [4].

A role-based access control model (RBAC) is based on the definition of roles in a certain system [123]. The link between the IT infrastructure and the real world is setup by the requirements engineering process. After that, the needs of the system are determined. Consequently, the process of so-called role engineering is executed, which consists of developing the scheme of a role in the IT world, the available permissions and the relation between the roles and the permissions [58]. The system administrators, which are person inheriting the roles, who control the security policy of the system, divide every user-object U into a minimum of one or several roles or user-groups [123].

An example for a role would be a "Health Care Provider". The user "Dr. Pink" would fall into this role. As "Dr. Pink" could also be a patient, he would gain the rights of a patient as well. The role "Health Care Provider" would probably have the allowance to add a new anamnesis, diagnosis or treatment object for one of her patients. Conversely, the role "Patient" would have the right to read her medical datasets.

3.6.2 Clinical Information Systems Security Policy

In 1996, Anderson published his principles regarding the secure handling of sensitive medical information [7]. Besides the standard demands of access control models, for example, that any object has to be secured with granular access rights and appropriate measurements have to be introduced to control every action undertaken on such an object, he specifies the tailored needs for medical systems.

Basically, he divides all medical data into three groups:

- Firstly, some information exists, like for example the blood group, which is known to all of the medical staff.

- Secondly, most of the data, which is gained by regular medical examination, is known to people in the practice and sometimes also shared with other health care providers. For example the results of an x-ray in an hospital will also be submitted to the patient's general practitioner.

- Finally, most patients have a subset of highly sensitive information, like a treatment for depression or an abortion. Hence, this information will be restricted to a single or very few persons.

This enumeration also conforms with the statements of the European Union's Article 29 Working Party. In their opinion, a patient herself may declare which part of her medical record may fall into which group [44]. Nevertheless, it is necessary to have a rule set, which automatically pre-selects the groups to minimize the administrative effort.

Following Anderson, the health care provider should be in charge of authorizing and revoking tasks. This is also the opinion of the US, which in contrast to the EU makes the health care provider and not the patient, the owner of the medical data. Though, the patient should also be informed about access to a specific record and planned changes to the users access list. For the latter, the consent of the patient has also to be obtained, if it is not an case of emergency [7].

Many laws in developed countries already demand long-term archiving of medial information. Upon introduction of the EHR this legal situation does not change. Thus, it should not be possible, that a health record is deleted without the patient's request [44].

Besides the confidentiality of medical records, internal security of an EHR system is a vital factor. Anderson not only demands excessive logging of access but also a notification, to inform a patient if another user gets access to large partitions of her data. The latter statement can also be broadened to include the secondary usage of medical data for research purposes [44, 45].

Hence, closely linked to access control is auditing, which means that various operations such as successful and unsuccessful access or log on attempts can be recorded in order to trace back specific users. It is possible to specify for each object which operations by whom should be recorded. Clearly, the integrity and confidentiality of the resulting log files are of utmost importance. No one should be able to modify (i.e. forge) log files and only system administrators should be able to read or delete them.

Summary In the previous sections we introduced the term security and its attributes. We stated cryptographic techniques to assure confidentiality and integrity. Moreover, we presented

principles, methods and workflows to realize secure systems. In the upcoming section we discuss additional security needs for medical systems.

3.7 Security and Privacy in e-Health

We already outlined, that firstly any architecture with the goal to realize the EHR has to deal with patients and health care providers needs. Secondly, appropriate measurements for assuring security and privacy have also to be established. Legal acts in several countries as well as personal opinions may influence decisions of stakeholders regarding the storage and usage of medical information. Nevertheless, the goal of an EHR architecture should also be to maximize the patients' privacy.

3.7.1 Hippocratic Databases

Regarding the secure storage of medical information, Agrawal et al. announced the following ten principles [3].

1. As sensitive information is kept in medical databases, the purpose for storing has to be specified to avoid misuse. Furthermore, a patient's anamnesis, diagnosis or treatment dataset needs to be associated with this defined purpose.

2. The patient has to consent every purpose of all stored personal information.

3. Besides the limitation of the data storage to specific purposes, only necessary data shall be collected.

4. Only queries against the database which are consistent with the defined purposes of the collected data are allowed. In other words, all access to the data with unspecified purposes has to be denied.

5. No personal information shall be disclosed beyond the consent given by the patients. Following this principle the secondary usage of medical data, is not allowed until the patient gives her explicit consent.

6. The storage of medical data needs to be limited to the time which is necessary to fulfill the specific purpose. This principle influences the creation of an EHR system. It is difficult to decide which datasets need to be retained over life-time and which medical data can be deleted. Health care teams need to establish a rule-set to determine which data needs to be stored over what time period.

7. To avoid misapprehension of a patient's medical history, all information in a data held in an EHR system shall be accurate. Moreover, all personal information needs to be checked and updated regularly.

8. Another vital point of a medical storage is the assurance of appropriate security measurements to prevent the misuse of the patient' sensitive medical data.

9. The participants of such a system have to be able to access all of their personal data at any time.

10. All users shall be able to verify if all of the principles aforementioned have been followed. This means that a medical storage system has to provide a compliance mechanism.

As sensitive medical information is stored within an EHR system, these principles assure that the patient is under full control of her data. Moreover, it limits the primary and secondary usage as well as the storage to specific purposes.

3.7.2 Risks in e-Health

> "[...] to do good or to do no harm [...]"
>
> *Hippocratic Epidemics [10]*

In 2004 Croll et al. invented a risk-based model regarding the appliance of an EHR system [35]. Basically they define (i) quality (q), (ii) usability (u), (iii) privacy (p) and (iv) safety (s) as as the vital key factors of an EHR implementation. Figure 3 shows the interdependencies between these different factors.

Quality can be seen as a factor which derives from fulfilled requirements and the system's robustness [35]. Moreover it adds value to a product and shows the conformance to specifications [105]. Especially in e-Health applications quality means the avoidance of harmful outcomings for the patients [35].

Following Nielsen [90] usability is a subset of the field of system acceptability. In other words, usability is "the question whether a system is good enough to satisfy all the needs and requirements of the users and other potential stakeholders" [90].

In case of medical systems, there are several groups of people involved, who need access to the patient sensitive information. Firstly, the patient and people like custodians or relatives have full access to the particular patient's data. As discussed in the previous section, only these persons

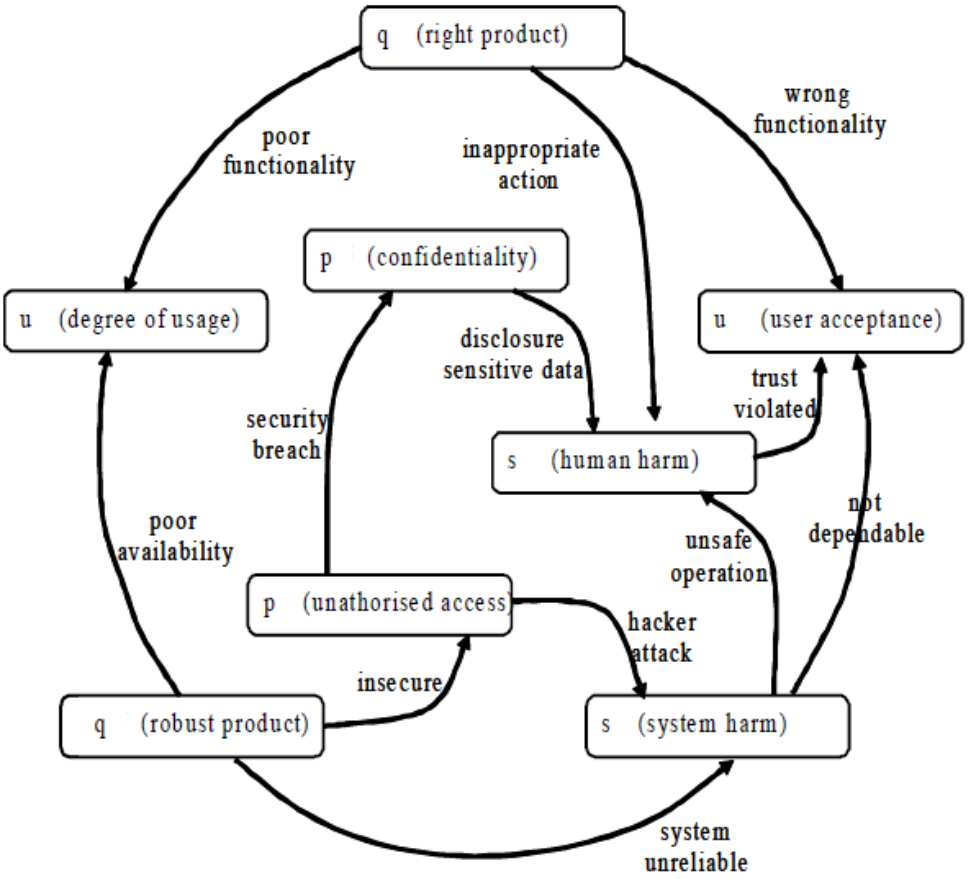

Figure 3: Risks in Medical Systems [35]

have the right to permit others with access to the patient's medical data. After a given consent health care teams [100] may read, write or modify the patient's health records. Commonly, these teams consist of general practitioners, other specialized physicians and their assistants. Moreover, health care providers and administrative staff in hospitals provide medical services for the patients [20]. Other potential stakeholders in medical infrastructures are pharmacists and employees at social insurance companies. Latter do only need aggregated anamnesis, diagnosis or treatment data to fulfill their tasks.

In order to meets the requirements of all these stakeholders an EHR system needs to offer various interfaces and applications. Thus, the implementation of such a system involves both, communication and application security [19]. Besides the trust in the system which can only be established by a working security model, a high system acceptability can only be achieved by providing appropriate usability.

As we already discussed in Section 2, privacy is a major concern of patients and data commissioners. It is a necessity to guarantee the patients' privacy and strengthen the confidentiality in

an EHR system, because privacy invasions directly lead to a decrease of trust. Thus, following the principles of the previous section, patients may not give their consent on sharing data with other health care providers. This may hamper the collaboration of health care institutions.

The fourth attribute presented by Croll et al. is safety [35]. All EHR implementations have to protect the patients' safety and avoid harmful outcomings for the patients [67]. For example wrongly entered data by health care providers or willfully changed data by attackers could lead to serious injuries or death. Though, if appropriate security and privacy measurements are taken, the EHR can help to decrease the number of adverse drug events (ADE). Regarding a study in the USA, ADE is the 8th leading cause of death and thus ranks for example before motor vehicle accidents or AIDS [67].

3.7.3 Medical Infrastructure

In this section we outline the users, organizations and systems involved in an EHR system (cf. Figure 4).

A hospital information system (HIS) is the aggregation of all data, components, workflows and software used in a clinic. In 2001, Kuhn et al. broadened the definition to from hospital to health information system [68]. This also reflects the need for preventive medicine in the health care sector.

Any HIS is based on a Picture Archiving and Communication System (PACS). This archive is most of the time split into a short-term and a long-term archive because the costs for on-demand data is rather high. Nevertheless, the majority of the countries demand storing of medical data for several decades. Moreover, the invention of the EHR relies on life-long storage [60].

Remark: In contrast to cost-intensive commercial solutions, the clinical image and object management software dcm4chee[9)] is available as open-source product. Thus, it is often applied in medical studies.

The workflows and dataflows are operated and controlled by a so-called Collaboration Server. This server is connected to the PACS and the shuttles of each health care provider. Besides health care providers' workstations, picture delivering units called modalities are used to observe medical information of patients.

A modern health care system is also based on standards to ease interoperability. For radiology

[9)] `http://www.dcm4che.org/`

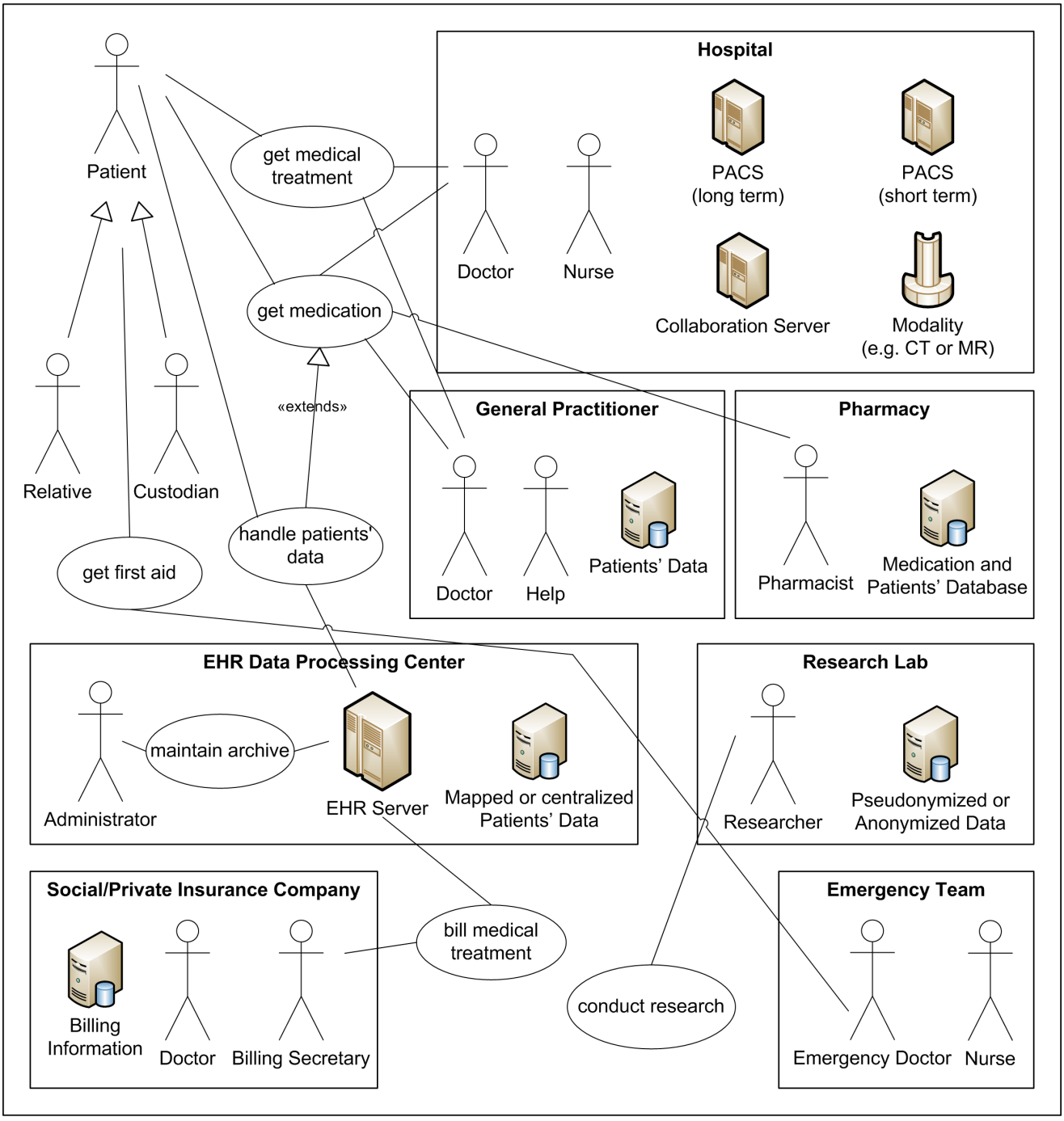

Figure 4: Medical Infrastructure

images, the Digital Imaging and Communications in Medicine (DICOM) Standard[10] is used. Nearly every available modality, like a magnetic resonance (MR) tomography unit or a computerized tomography (CT) scanner delivers images in that format. For inter clinical exchange

[10] `http://medical.nema.org/dicom/`

the Health Level 7 (HL7) Standard[11)] emerged. The current version is already based on the semi-structured W3C XML (Extensible Markup Language) schemes.

Commonly, general practitioners and other specialized physicians are equipped with medical systems, which offer a similar functionality as a HIS, but are limited in their interoperability [35]. The connection of such systems with a nation-wide EHR can also be realized by the usage of HL7.

There are three possibilities to exchange data between different health care institutions. Besides a centralized data repository holding all medical information, a centralized database can be setup in order to support collaboration. Moreover, some countries will implement a mixed structure, in which parts of the data is hold centralized but most of the access to the medical information will be realized by linked archives [44]. For example in Austria, the data itself will not be exchanged between the different health care institutions. Instead the government plans to setup a system which will only link the different databases [60].

Pharmacists are another user group which has access to an EHR system. In contrast to health care providers, who are permitted to view or change a complete medical dataset, pharmacists do only need information about the prescribed medication. Thus, e-Prescription is another requirements of a nation-wide health care system.

Social insurance or private medical insurance companies handle the financial tasks within an EHR system. They need to control whether a certain treatment is covered by their policies. Hence, these organizations have to be able to access at minimum the aggregated anamnesis, diagnosis and treatment data.

Besides the regular access of patients, health care providers, pharmacists and employees of insurance companies on the medical data, an emergency access needs to be realized in order to protect patients' health. This type of query against the data differs, because a patient may not be able to give a consent if she is unconscious. Thus, every patient should be able to define a subset of her data as emergency data [96, 111, 141]. As this information may also hold sensitive information, appropriate privacy, security and safety needs to be assured. To avoid misuses and assure patients' control, it is no only necessary to define the emergency doctors as a user group, but also to log all access to this data.

In contrast to the primary usage of health data as aforementioned, medical research institutions contribute to the improvement of the quality of treatment by conducting medical studies. Secondary usage also helps to develop better medication [99]. From a security point of view,

[11)] `http://www.hl7.org/Library/standards.cfm`

anonymization (cf. Section 2.2.1) or pseudonymization (cf. Section 2.2.2) of the patients' medical data are privacy-enhancing technologies to preserve the patients' privacy.

Regarding the workflow within an EHR system, the research institution firstly chooses the parameters for the query on the medical data. Secondly, the patients are either questioned by the system or the general practitioner if they want to participate in the study. If the patients give an explicit consent, the data will be anonymized and afterwards released. To ease data exchange, the system of the research center can also retrieve the data via HL7.

Summary In the last sections we introduced different additional constraints of security and privacy in health care systems. The major goal of every EHR system is to assure the safety of its users to avoid injuries or death of people.

We further discussed how data is stored and handled in medical environments. Standardized protocols like HL7 may help to build interfaces for the connection of the various existing systems with the EHR.

Several EHR architectures, emerged which try to provide a secure environment for the primary and secondary usage of medical data. Nevertheless, most of them have security or efficiency drawbacks [107, 109, 110, 112–114].

3.8 EHR Architectures

An EHR not only has to solve the primary usage of medical data for patients' treatment, but also helps increasing quality of medical care by the secondary usage. The latter type may be based on anonymized or pseudonymized data depending on the conducted research. Examples for secondary usage purposes are on the one hand disease specific clinical and epidemiological research projects and on the other hand health care research, assessment of treatment quality and health economy. Typically for this type of use, the identity of the patient does not influence the results [99]. Thus the appliance of aggregated data may lead to the same results [25].

The most secure method for secondary usage of medical data is total anonymity (cf. Section 2.2.1), because this approach eliminates the link between the anonymized data and its associated individual. Thus, it is the most secure way for granting privacy. Although this approach is often used in research projects due to its simplicity, it has the major drawback that patients cannot be informed about actual findings of a certain study such as new developed medical treatment or major changes in the healing progress.

In the upcoming sections we discuss current approaches towards a secure EHR solution. Then we work out several principles for medical-related privacy-focused architectures in Section 3.9. These principles can be seen as guidelines for the design of new EHR architecture or the upgrade of an existing system to a PAT.

Different solutions have been published to securely conduct primary and secondary usage of medical information. In the following, we investigate different EHR architectures, especially regarding their security and privacy implementation.

3.8.1 Thielscher Architecture

Several approaches have been contributed to provide a secure architecture for establishing an EHR. The system published by Thielscher et al. [141] is based on decentralized keys stored on smart cards.

Their approach consists of two databases, one for the patient's identification and one for the medical data. The relation between a certain patient and her datasets can only be established by applying the necessary secret key on the smart card in order to generate the unique data identification code (DIC). None of the DICs can be associated with a particular patient. In other words, the applied pseudonymization technique assures that attackers who steal parts or the complete patients' medical database, are not able to invade the patients' privacy.

The system allows to authorize health care providers (HCP) to access specific anamnesis, diagnosis or treatment datasets by sharing one or several DICs. An optional security aspects is that the authorization is limited by time. This means, that the health care providers' access to the data is restricted to a certain time period after the last successful authorization by the patient.

Another property of Thielscher's system is the procedure in the case of a medical emergency. They propose that an emergency call center is able to access the patients' data after an health care provider has authenticated herself.

The major shortcoming of this system is the dependence on a centralized patients-pseudonyms list, which provides a fall-back mechanism in case a patient loses her smart card because otherwise there would be no possibility to recover the mentioned identifier. Thielscher et al. circumvent this security flaw by operating the patients-pseudonyms list off-line.

This organizational work-around promises a higher level of security until a social-engineering attack is conducted on a system's insider [77, 144] or an attacker gets physical access to the computer that holds this list [107–110, 112–114].

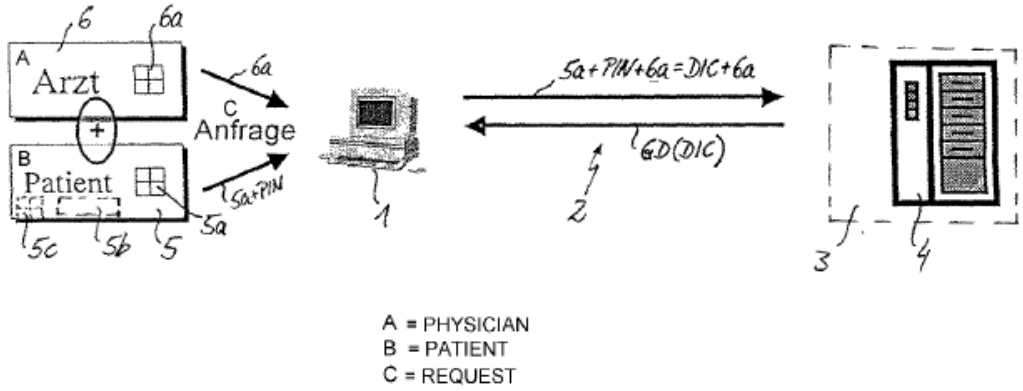

Figure 5: Thielscher Architecture [141]

3.8.2 Pommerening Architecture

Pommerening et al. contributed two different approaches [98, 99], which are both similar to the system of Thielscher et al. (cf. Section 3.8.1). Their architecture, which is only applicable for the secondary use of medical data in research centers, is a combination of a hash and an encryption technique.

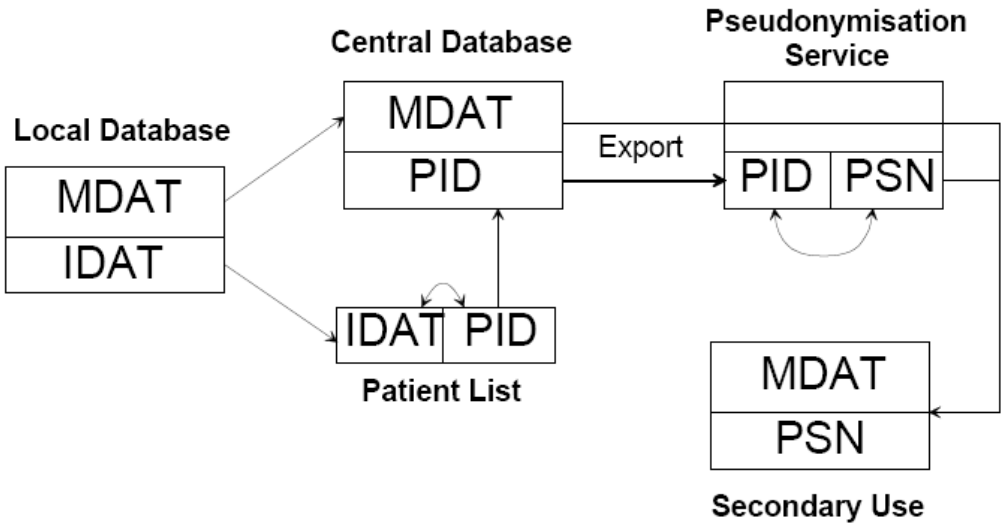

Figure 6: Pommerening Architecture [99]

Their basic workflow consists of the following steps: Firstly, the identification data (IDAT) is separated from the medical data (MDAT). Afterwards a hash algorithm, which they call PID Generator, is applied to form a unique identifier (PID). Then this identifier is encrypted by a pseudonymization service to calculate the patients' pseudonyms (PSN). The last step is that the medical data together with the pseudonyms are made available to research centers for secondary

usage. [98, 99]

In case the pseudonymization needs to be reversed, Pommerening et al. use a patient list [98,99]. The encryption itself is based on a centralized secret key, which opens a vulnerability, because if an attacker knows this single key, she might gain access to all patients' related medical data. Moreover, it is also possible to compromise the patient list too.

Another attacking possibility would be that every user's PID and the resulting PSN is only unique for every single patient and not for every patients' medical dataset. Hence, it is possible to conclude the medical history by combining all pseudonyms of a particular patient. As a consequence the identity of certain patients may be guessed on the basis of this patient's medical history. From a security point of view, it would at least be necessary to assure that only within one medical study, the same PSNs are used to avoid these types of attacks.

Currently, their system is a vital part of several research centers.[12)]

3.8.3 Peterson Architecture

The approach of Peterson [96] relies on the usage of different keys and corresponding passwords and is also based on a centralized table, which is used for re-identification purposes. Regarding the security, every patient holds two unique keys: One that is only known to the patient (personal key, PEK) and the other one is printed on a card (global key, GK). Both may be used to access the stored information. Editing of data is only allowed if the patient's password is available too.

Peterson argues that "[...] the unpredictable nature of medical emergencies means that the need to know for essential medical information could be anytime of day or night, and anywhere in the world.". And further "Typically, the GK will be a long alphanumeric string, which is difficult to remember [...]".

In our opinion, attacks on the patient's global key are possible even if the key can hardly be remembered because one can, for example, make a copy or photo of the patient's identity card. Moreover, the aforementioned table is, from a security point of view, comparable to the approaches of Pommerening [98,99] or Thielscher [141] and therefore relies on the same weak point, because a centralized list is attackable from in- or outside the system. In other words, it is a promising goal for any attacker.

Peterson also proposes the encryption of all data with a server-side key (SSK). As medical data tends to be very large (e.g., the image size of a x-ray is 6 MB, for a mammogram 24 MB or for a

[12)] Telematikplattform fuer Medizinische Forschungsnetze (TMF) - `http://www.tmf-ev.de/`

Global Key Available	*Personal Key Available*	*Password Available*	*Resulting Action*
No	No	No	Access denied
Yes	No	No	View only
No	Yes	No	View only
Yes	Yes	No	View only
No	No	Yes	Access denied
Yes	No	Yes	View and Edit
No	Yes	Yes	View and Edit
Yes	Yes	Yes	View and Edit

Table 6: Peterson Architecture Use Case Scenarios [96]

computer tomography scan counts up to hundreds of MB [1,84]) and encryption is a highly time consuming operation, encrypting all data would not be feasible [110].

3.8.4 Onuma Architecture

Onuma et al. [94] invented an architecture (cf. Figure 7) for the secondary usage of medical data. It comprises of the de-identification process of the sensitive information.

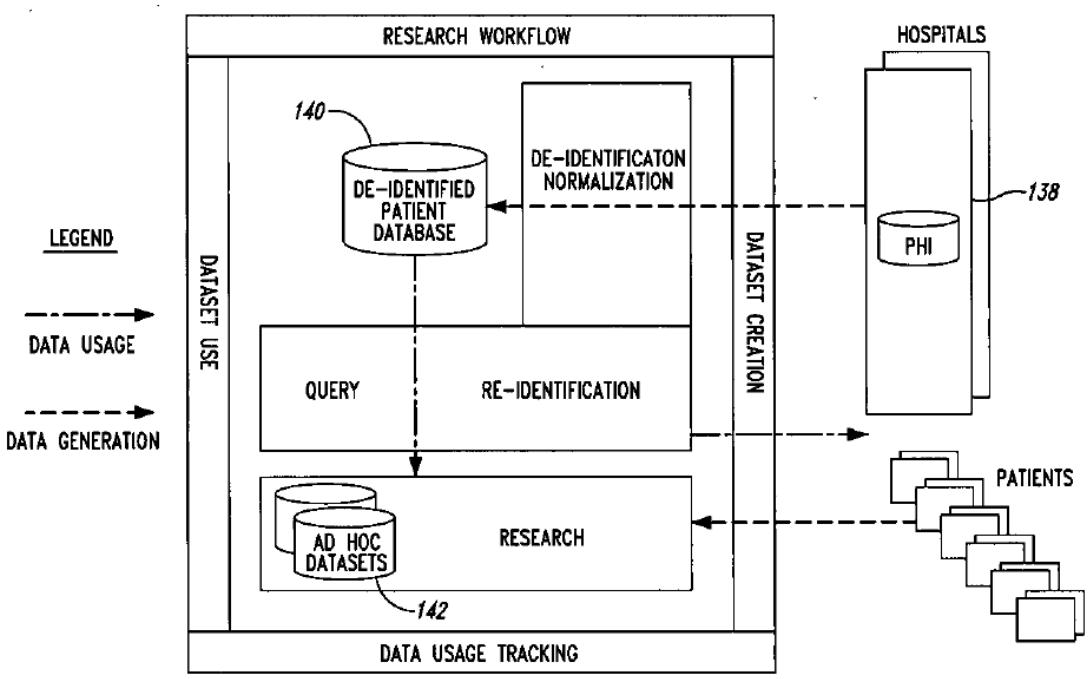

Figure 7: Onuma Architecture [94]

Firstly, the data will be normalized to meet the requirements of their storage. Secondly, all identifying attributes will be removed from the patient health information (PHI) by the usage

of a de-identification agent. Afterwards, the identifiers will be replaced with a unique identifier which is based on the appliance of a key, which can be random. This key will be stored in a centralized keystore.

Onuma proposes that the authorization for screening the patients' data for secondary usage in a medical study, can be realized for example by an internal review board. This board decides about the possible usage of anamnesis, diagnosis or treatment data based on the query and characteristics of the study. In addition Onuma et al. propose that the selected attributes in the query may be previously stored in the de-identified medical data.

If researchers who work with the data see the need to re-identify a certain person, a re-identification module unveils the relation between the patient and her data by applying the appropriate key. The centralized keystore is secured with means of access control and the usage of an authorization processor.

As previously mentioned, access control models should only be used if encryption is not applicable. Moreover, the security design principle of separation of privilege (cf. Section 3.4.6) could help to enhance the security of the centralized keystore. In that case the decision of the internal review board also triggers the change of reading permission on patients' data, but in addition the cryptographic means of for example a threshold scheme would increase the operational security [131].

Equivalent to the approach of Thielscher et al. (cf. Section 3.8.1), Onuma et al. outline the usage of restricted time periods, in which authorized persons, for example researcher, may access the data. Nevertheless, this security enhancement is realized by the usage of access controls in combination with log files instead of cryptographic protocols.

The security of de-identification protocols for data privacy is mainly based on the quantity of the stored data. As aforementioned, the more data exist, the harder it is for an attacker to unveil the relations between users and their data based on a profiling attack. Further, the goal of the implementation of such systems is to realize centralized data repositories. Both factors lead to the conclusion that a huge number of permitted access operations on the data will create enormous log files. Thus, monitoring of these log files will create an additional effort for the system maintenance and can therefore hardly be realized.

3.8.5 Schmidt Architecture

In 2001 another architecture was proposed by Schmidt et al. [126]. The information is stored in two different databases. One database holds the identification data, whereas in the other the

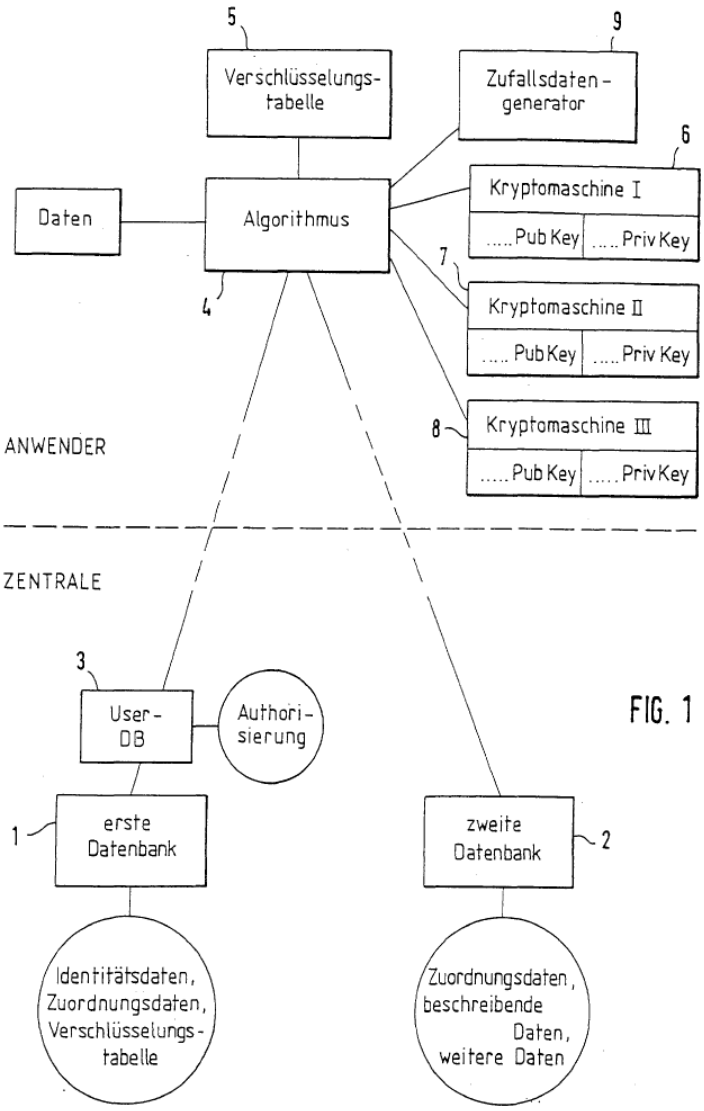

Figure 8: Schmidt Architecture [126]

medical information with an assigned date will be saved.

The relation between the two databases is concealed, because this assigned date is stored encrypted. To re-establish the relation between the patient's identification and medical data, a user has to belong to the assigned users group which are permitted to access the specific anamnesis, diagnosis or treatment.

Every user of this group will be provided with the associated key pair DomPrivKey and DomPubKey. The public key, DomPubKey is preliminarily used to encrypt the specific file's secret, the FilePrivKey. The file's public key FilePubKey has been used to conceal the file's identifier.

Schmidt et al. propose the usage of a symmetric file key FileSymKey as better performing

alternative. Anyway, the user group's private key has been encrypted with all of the authorized users' keys to secure access. For that reason, every participant holds a public (IndPubKey) and a related private key (IndPrivKey). Hence, only users which possess the necessary keys are able to gain access to the assigned date by step-wise decrypting the applied keys.

The concept assures an appropriate level of security because authorizations are given by sharing encrypted secrets. Nevertheless, as the DomPrivkey is shared amongst several users, and Schmidt et al. explicitly state that every participant will be equipped with these keys after a successful authorization, revoking is difficult.

Firstly, a user might write down the key and is therefore still able to use it after she has been revoked, without the knowledge of the data owner. Secondly, a user could share her knowledge and authorize other users without the data owner's permission.

Hence, to avoid misuse of old privileges in case a user should be revoked, another authorized user has to decrypt the DomPrivKey by the usage of her IndPrivKey. Further, she has to use the plaintext DomPrivKey to decrypt the FilePrivKey to finally get access to the file's identifier. Afterwards she would have to choose a new FilePrivKey / FilePubKey pair and in addition and new DomPrivKey / DomPubKey. The new encrypted identifier is then a result of the encryption of the file's identifier with the FilePubKey.

To assure continuous access of all already authorized users but the one who should be revoked, an encrypted version of the FilePrivKey has to be issued to the holders of the DomPrivKey / DomPubKey combination. As all users possess different individual IndPrivKey / IndPubKey pairs, the dedicated authorizing person needs to ask all participants for their public keys.

In our opinion this process could be eased by the usage of either distinct keys or additional differencing attributes for every user and file combination. In that case, only the particular file's encrypted identifier of the user who has to be revoked needs to be removed from the dataset.

Moreover, the approach of Schmidt et al. misses out a reliable backup-concept for the used keys. Again, we want to emphasize the usage of a threshold scheme to minimize insider attacks.

3.8.6 Slamanig and Stingl Architecture

In 2008 Slamanig and Stingl proposed an e-Health PET for storing reference data instead of the documents themselves [133]. Hence, their focus lies on the authorization concept and not the data storage. As already discussed in other architectures, they divide the underlying data repository into two database, named user repository and document repository (cf. Figure 9).

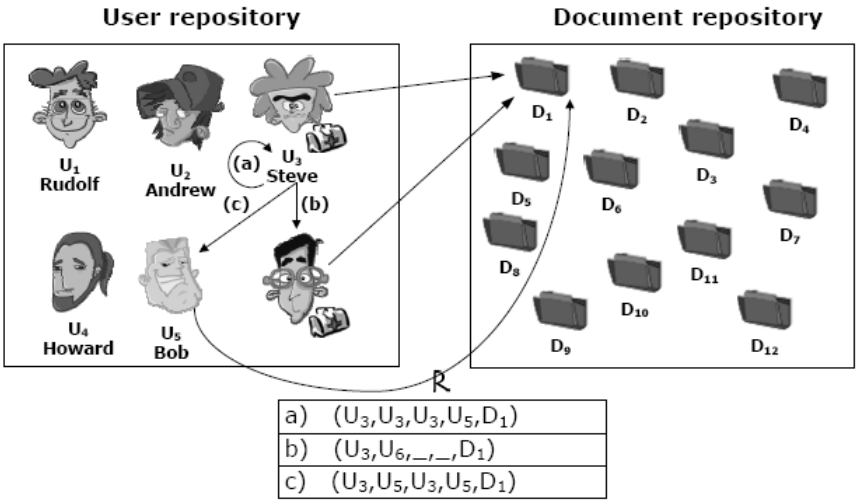

Figure 9: Slamanig and Stingl Architecture [133]

An access right is represented by a five-tuple (U_S, U_R, U_C, U_P, D_j), whereas a user U_S permits another user U_R with access to a patient's U_P document D_j. User U_C has been responsible for the creation of this authorization. Any access to one of the documents is secured by the usage of a distinct key. In addition, the attributes of the authorization tuple are held encrypted.

Slamanig and Stingl also emphasize the use of an identity management to avoid a so-called disclosure attack. As a particular EHR holds all of the patient's medical data, a permission to the EHR may also lead to the disclosure of all stored medical data. In their approach identity management means the creation of k sub-identities, or pseudonyms of every person. On the one hand, from a security point of view an attacker may then only gain unauthorized access to a subset of the patients sensitive data. On the other hand, it is difficult to determine how these sub-identities should be established.

Moreover, they discuss the possibility that employers may treat for example persons who suffered from a burn-out syndrome differently. The approach of establishing sub-identities as an organizational aspect might not change this situation at all, because employers can still demand the disclosure of all medical data or otherwise refuse employment. We think that the legal situation of this issue cannot only be solved by technical means, but it also has to be handled by legal acts.

In the last sections we discussed different approaches regarding their security and privacy constraints. With the outcoming of our secondary literature study and the basics introduced throughout this section we now work out ten principles for the secure pseudonymization of medical data.

3.9 Principles for Medical-related Privacy-focused Architectures

Based on the security techniques presented in the previous sections, we define the following ten demands for a system that allows the secure pseudonymization of health care records: [107, 109, 110, 112–114]

1. Use depersonalization to separate all data stored into two different tables or databases [104], one for the personal data and the second one for the medical data. This provides the basis for applying pseudonymization as confidentiality [9] technique.

2. Replace the foreign key in the medical database, which is related to specific persons, with a pseudonym [73, 139] to assure the patient's privacy.

3. To reduce the possibility of profiling attacks on the medical history and excessive key changing protocols, every combination of patient, health care provider and medical dataset should be defined with a unique pseudonym [107, 109, 110, 112–114]. Moreover it is important to hide any relation between interacting persons for the same reason.

4. Secure the keys to form pseudonyms and not the algorithm as demanded by Kerckhoffs' principle [128].

5. Apply a threshold scheme, to share secrets like keys [131]. Moreover, conceal the association between the patients and their responsible administrators. This demand assures that not one single person is able to unveil a certain person's identity.

6. Following the previous point, there is the requirement to balance the number of administrators [131], which are assigned to hold a certain person's backup key, and the number of administrators, which are necessary to act together to unveil the secret.

7. Use role-based access control models only as a layer above the technique of giving permissions to datasets by sharing encrypted secrets (for example keys or hidden relations) to access certain pseudonyms because this is the highest security level for authorization purposes [107, 109, 110, 112–114].

8. Provide the patient with the possibility to decide which datasets she wants to share by forming an unique pseudonym for the patient herself as well as for any patient-health care provider-medical data combination. In addition hand over all rights to authorize or revoke persons, as far as possible and legal [34, 43, 106, 147], to assure that the patient is in full control of her data.

9. Regarding the secondary usage of anamnesis, diagnosis or treatment data, the patient has to consent the appearance of her sensitive information in medical studies. Measure the k-Anonymity not only but especially for follow-up studies to avoid re-identification attacks on the patient's data [71, 121, 135, 136].

10. Apply appropriate means of control for the patient so that she is able to view all stored data and permitted authorizations on her medical data [3].

Based on these demands and the security constraints presented in the last section we introduce the description of our system in the following section. This architecture is the basis for the set-up of our novel formal workflow framework in Section 5.

4 System Architecture

In this section we sub sum the requirements for a secure e-Health architecture and afterwards introduce our architecture PIPE (Pseudonymization of Information for Privacy in e-Health). PIPE can be used for the primary and secondary of medical data. Moreover, due to its layered architecture, it could work as a PAT in existing applications or be the basis for new implementations.

4.1 Secure Pseudonymization

Pseudonymization is a technique where identification data is transformed into a specifier and then afterwards replaced by that specifier. It is not possible to associate the latter with the identification data without knowing a certain secret [97, 109, 112, 114, 139]. As it is necessary for privacy reasons to avoid storing any personal information with the pseudonymized dataset, a pseudonymized database has to contain at least two tables, one where all the personal information is held persistently, and another one which keeps the pseudonyms and the pseudonymized data.

After depersonalization and subsequent pseudonymization, a direct association between certain persons and their data cannot be established. Algorithms for calculating the pseudonym may be based on encrypting or hash techniques [73]. The latter demands to store a list where all pseudonyms are kept in order to assure reversibility [18, 48, 98, 99], but relying on the use of a list is not secure, as an attacker, who gains access to this list, could establish an unauthorized relation between the identification and the medical data of a specific patient [107, 109, 110, 112–114]. Encryption provides a more secure alternative for building pseudonyms. For using encryption with a symmetric algorithm, a secret key or for the asymmetric alternative, a key-pair, is needed.

As demanded by Kerckhoffs' principle [128] only the keys have to be kept secret, whereas the applied algorithms are accessible. Hence, a major requirement for a secure system is that keys have to be shared with as few persons as possible, preferable with nobody. Nowadays it is a common practice to store keys on smart cards [59, 103]. They are equipped with a small logic chip in order to conduct cryptographic operations without the need to process data on open systems like a standard client, for example a personal computer.

Hence, this technique in combination with a certified card reader assures confidentiality and integrity [9] of sensitive data while encrypting or decrypting. In other words after authenticating against the smart card by entering a PIN, data is transfered to the card-reader and afterwards processed on the card's crypt chip. If the PIN is only accessible to the card holder, this technique can be considered secure [59, 103].

However, as smart cards may be lost, stolen, destroyed or compromised, it is a system's require-

ment to provide a fall-back mechanisms that allows recovering the key in order to re-establish access to the data which has been encrypted with the smart card. Hence, all keys should be kept centralized within the system in a backup keystore which needs to be secured itself.

Role-based access control models could be used for handling the authorization and authentication tasks for the backup keystore, but as role-based access control models can be by-passed or compromised [119,154], a high level of security can only be established by encrypting the keystore itself [114]. Nevertheless, persons with administrative roles have to be granted access to the backup keystore for maintenance purposes [114, 141]. Therefore, this technique does not provide enough security for sensitive health data, because attacks could be performed by or through people working inside the system, e.g. by social engineering attacks [144]. In order to mitigate this vulnerability, threshold schemes [131] allowing to share keys in the backup storage between multiple administrators can be used.

Another shortcoming of existing systems is the patients' dependence on a single pseudonym. If a patient only holds one pseudonym, an attacker who gains access to the database could conduct a profiling attack on the patient's medical history to re-identify a certain the patient. For example, only a group of patients might have had a knee operation in a specific time slot. Moreover, perhaps only a few people of this group had been treated at a certain hospital and only one of them has seen her dentist a couple of days around the knee operation. Hence, as it is possible to conclude the identity of a person by combining single occurrences of her medical data, the usage of pseudonymization can only be considered secure if enough disjointed pseudonyms exist.

In the next section we provide a detailed view of the functions and permissions of the different roles and components of our architecture.

4.2 Roles and Components

The goal of our architecture is to gain the optimal trade-off between maximum security on the one hand and usability and performance on the other hand. Relying on encryption techniques for authorization purposes allows us to establish a higher level of security.

We outline the roles and components in our architecture. Then, we continue with a presentation of the design principles and security methods applied.

Our architecture (cf. Figure 10) consists of the following users $\mathcal{U}$ and components:

- A central system (e.g. server, etc.) which provides access to a central storage (St) which itself is divided (e.g. logical, physical) into two separate storage systems (e.g. databases,

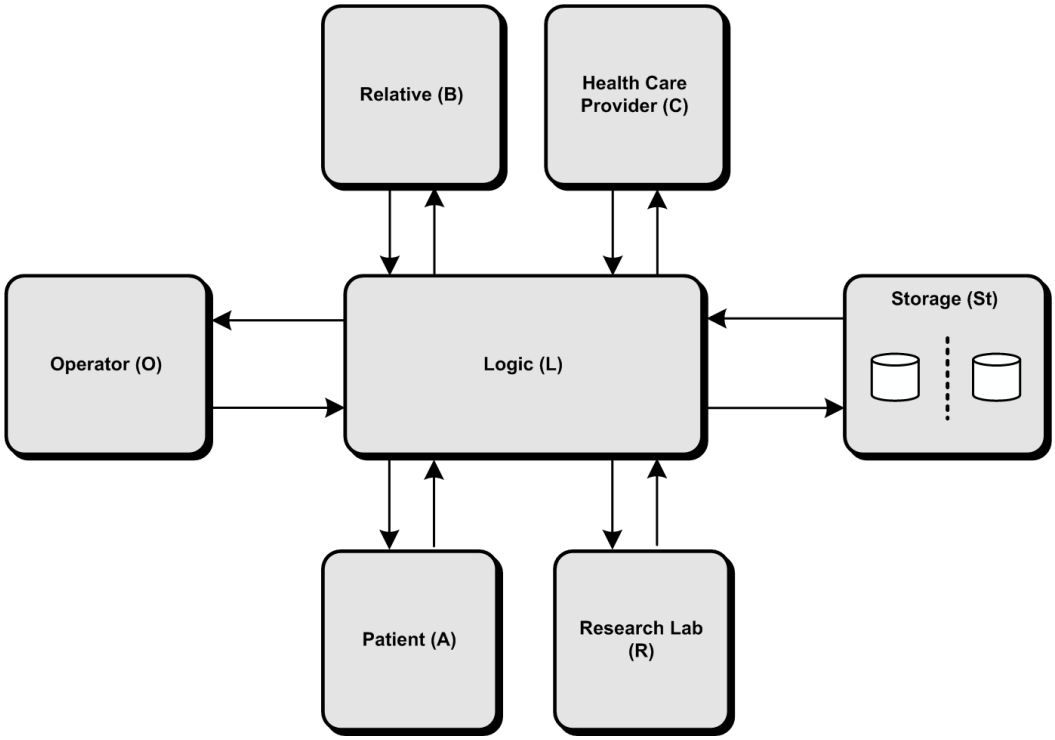

Figure 10: Architecture

etc.), where one is related to identification data and the other one is related to data, which should be pseudonymized as well as the associated pseudonyms,

- a central logic (L) that provides an interface between the central storage and the clients for the purpose of saving and loading the data,
- the patients ($\mathcal{A}$) who have full access to their data on the central system via the central logic by using a security token (e.g. smart card with a PIN, biometric authentication, etc.),
- the relatives ($\mathcal{B}$) who might get the same rights as the patient by default, if not supervised by a role-based access control model,
- the health care providers ($\mathcal{C}$) who share one or several entries in the pseudonymized database with the patients,
- the research labs ($\mathcal{R}$) that have solely access to the anamnesis, diagnosis or treatment data on the server system via the central logic for the purpose of analysis needed for improving the efficiency of clinical trails, the medical treatment, or medication and
- the operator(-team) ($\mathcal{O}$) which can hold secrets on behalf of the system. In other words this role assures that if a patient loses or destroys her smart card, the access to the system

can be restored by a team of operators.

In practice the logic L, and the storage St, which might be outsourced to a data processing center, have to form a trusted instance, because smart card management is handled there.

4.3 Authorization by Encryption

Our architecture is based on a layered model with a minimum of three security-hulls representing the authorization mechanism. Every hull includes one or more different symmetric keys or asymmetric key pairs. A key K_N of a certain hull H_N is encrypted with a key K_{N+1} of the hull H_{N+1} enveloping hull H_N. We identify each medical entry in the database, which represents the concealed data hull, by the usage pseudonyms, which are secured by the most inner key. Following the security design principle of economy of mechanism (cf. Section 3.4.3), this is the only possibility to access pseudonymized data stored in PIPE.

To establish a link for a certain patient and an entry in the concealed hull, a patient, or a system on behalf of the patient, conducts the following operations. The user starts by authenticating against her most outer hull security token, for example by entering a pin to authenticate against a smart card. The key of the authentication layer, which is, in our example, stored on a smart card, can be used to decrypt the encrypted secret key, in the next inner hull, the user permissions hull. After gaining access to this key, the patient selects a medical dataset and decrypts the key which is related to the selected dataset and thus, she is able to establish a link to the data by calculating the pseudonym again. Please note, that all encryption and decryption operations are transparent for the users to comply to the security design principle of psychological acceptability (cf. Section 3.4.8).

As role-based access control models may be compromised or by-passed [119, 154], we based all access and authorization techniques of the system on encryption as far as possible. In other words, authorization is given by sharing certain keys or secrets between users in the inner hull/user permissions layer. Authorized users encrypt the received key(s) or secret(s) with their own inner key to safeguard them.

From a security point of view, it is still possible to share given secrets with other users without the notice of the user who gave the original permission to share a certain secret. Hence, it is also a necessity to share secrets with as few users as possible. This strategy also conforms with the security design principle of least privilege (cf. Section 3.4.1).

Moreover, shared secrets may also be divided between more persons which have to act together to unveil a certain secret, to minimize the risk of misuse. Besides the usage of encryption for

authorization purposes, we implemented a role-based access control model on top of the security stack, which helps to prevent frauds and provides additional logging functionality to assure non-repudiation. We further developed a mechanism, which minimized key changes, because this would add unnecessary overhead.

As aforementioned we additionally developed a role-based access control model as top layer (cf. Figure 11) to refine the access rights and achieve access control beyond authorization on encryption. This also complies with the security design principle of complete mediation (cf. Section 3.4.4).

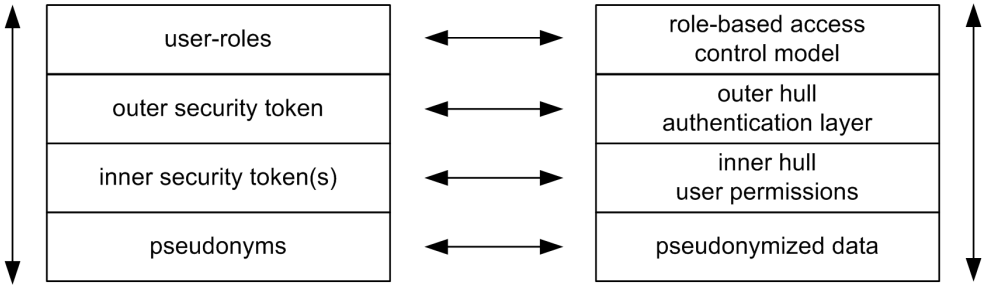

Figure 11: Security Layers

In the following we will introduce the rules of our role-based access control model. Please note, that we provide a summary of all security attributes used in PIPE in Section 5.

1. No operator is allowed to see the ciphertext of the patient's identifier encrypted with the logic's inner symmetric key $\{A_{id}\}_{\overline{K}_L}$ while processing the backup key operations to avoid informal agreements between the operators (cf. Section 4.5).

2. A health care provider needs to be authorized to have the possibility to create datasets for another user (cf. Section 5.3.1).

3. As the patient's inner private key $\widehat{K}_A^{-1}$ can be shared by the role relative, B would have the same rights as the patient. To avoid data misuse, a relative is only allowed to read all data, writing and authorizing is denied. Moreover, viewing of the patient's inner symmetric key is not permitted. Thus, it is possible to securely revoke an authorized relative by changing the patient's inner private key (cf. Section 5.3.2).

4. Only data owners who possess the so-called root pseudonym are able to conduct delete operations (cf. Section 5.3.3 and Section 5.3.6).

5. It is only possible for users, which possess the root pseudonym, to authorize (cf. Section 5.3.7) or revoke (cf. Section 5.3.8) other users.

4.4 Unlinkability and Unobservability

Besides the usage of encryption for authorization purposes, one vital point of our system is that the medical data itself does not have to be encrypted [107, 109, 110, 112–114]. Instead we assure privacy by securing the link between the patient's identification data and her anamnesis, diagnosis as well as treatment data. Hence, if health care providers store only medical data which cannot be related to a certain patient, unobservability is guaranteed. In other words, any attribute which could be associated with a patient has to be stored with the identification data and not with the medical data.

Nevertheless, it would be a simple attack, if the same pseudonyms are used repeatedly, because the risk of profiling attacks arises. Hence, we have to assure, that the patient's medical history cannot be concluded from her pseudonyms. In PIPE all combinations of patient-health care provider and medical dataset have a unique and distinct pseudonyms.

To gain unobservability, we need to conceal all actions undertaken by the users, as far as possible and complying to legal acts. Regarding the level of anonymity as discussed in Section 2.2.1, in a system holding personal medical data, a minimum of level 4 anonymity should be setup, because users need to be identified at least in cases of emergency. Still, only authorized users are allowed to relate the patient's identity with her anamnesis, diagnosis or treatment data. Thus, inside of PIPE's storage, we guarantee a level 3 anonymity. This means, that all medical data is pseudonymized, but can be revealed by a group of system operators. Hence, the parameter complexity of anonymity is set to the number of operators necessary to unveil a certain secret. This measurement decreases the risk of frauds.

Summary In the previous sections we introduced the cornerstones of PIPE. Another aspect of a secure EHR system based on PIPE's novel pseudonymization workflow framework is that, smart cards may be lost, destroyed, compromised or just worn-out. Thus, we present two possibilities of sharing an inner key for backup purposes.

The first possibility in our architecture is, that the inner key pair of a patient may be handed over to other users, e.g. a relative, a custodian or nursing staff. Nevertheless, this technique also grants access to all of the patient's sensitive data, which may be undesirable. Though, this concept assures, in case of a lost or destroyed outer key pair, that the secrets of the concealed hull are still restorable. Otherwise, there would be no possibility to gain access to the user's pseudonyms and thus the data would be lost forever.

We discuss the second possibility of sharing an inner key for backup reasons in the next section.

This technique complies with the security design principle of separation of privilege (cf. Section 3.4.6) and can therefore considered more secure than the first alternative.

4.5 Establishing a Backup Keystore

We already mentioned that the need exists to assure that users still have access to their data if they lose their smart cards. In our system we provide an appropriate fall-back mechanism by the possibility of sharing the user's inner private key, which grants access to the inner hull key and subsequently to the pseudonyms. For instance, a relative could hold another encrypted version of the user's inner private key. Thus, she would get the same access to the data, as the patient gets herself, if not controlled by a role-based access control model.

Maybe someone does not want to grant access to all data to a certain relative. Thus the demand arises to store a backup of the necessary keys inside the system. This would also ease recovering keys and issuing of new smart cards.

Due to the aforementioned security reasons, these secrets have to be divided between more persons. In our architecture, we applied Shamir's threshold scheme [131] (cf. Section 3.3.4) to divide the user's inner hull key into n shares. At least k of these n shares are necessary to reconstruct the whole key. The n shares are randomly distributed amongst all operators, which we therefore define as assigned operators. Moreover any assigned operator may only hold a maximum of one share of a certain user's key. Hence, k necessary operators for every user exist, which have to act together to unveil her key. This difference between n assigned and k necessary operators provides a fall-back mechanism because one operator could be ill or does not have a working smart card. Hence, it also raises the availability.

We named the set of assigned operators $\mathcal{O}^n \subset \mathcal{O}$ and the subset of necessary operators $\mathcal{O}^k \subseteq \mathcal{O}^n$. Following Shamir [131] it is not possible to combine $k-1$ shares to compute the key, but if an attacker is able to bribe $b \geq k$ operators, she may succeed in unveiling a certain user's identity. Hence, we state the probability of guessing the necessary operators for a specific user under the condition that the operators do not know for whom they are holding shares in Equation 4, which is hyper geometrically distributed.

$$P(k \leq X \leq n) = \sum_{\iota=k}^{n} \frac{\binom{n}{\iota}\binom{|\mathcal{O}|-n}{b-\iota}}{\binom{|\mathcal{O}|}{b}} \tag{4}$$

This equation leads to the following conclusions. The larger the group of operators, the lower the probability that an attacker could bribe the assigned ones to find out a certain patient's identity.

The lower the minimum of operators necessary to unveil the secret compared to the number of operators assigned to a certain patient, the higher the probability for a misuse of the system. If the operators do not know for which person they share secrets, an attacker has to compromise, in the worst case, all operators minus the number of assigned operators to get access to a secret of a specific person.

To conceal the information of which operator is related to a certain patient, the system encrypts the secret shares first of all with its logic's key and afterwards with the inner public keys of the operators. Hence only if an operator knows the logics's key it is possible for her to unveil the relation, but she needs more operators to rebuild the shared secret. Therefore, the possibility for arrangements between the operators is lowered.

In the next section we provide examples for several combinations of assigned and necessary operators and investigate their security regarding different system sizes.

4.5.1 Security Obligations with One Operator Type

Figures 12 and 13 present the behavior for a system with 100 operators and four different settings of assigned and necessary operators. Whereas Figure 13 shows the probabilities of accessing the system for a range of 0 to 100 bribed operators, Figure 12 shows an extract for the range of 0 to 20 bribed operators. In our opinion the latter one is more important, because if appropriate organizational measures are applied, it should not be possible to bribe a large number of operators. Such measures could be for instance security handbooks to counterfeit social engineering attacks (cf. [144]).

Shamir stated, that a minimum of $n = 2k - 1$ users, in our case operators, required to unveil a certain secret makes a "very robust key management scheme" [131]. We therefore define all four pairs with this constraint. From an economical point of view, the combination of 2 *necessary* and 3 *assigned* operators results in the minimal costs for applying the threshold scheme with the minimum fall-back of one backup operator. Note, that the higher we set the number of necessary operators, the lower the quantity of users they are able to serve, because it consumes working time to handle recovering key requests.

Following Shamir, a security model with 3 *necessary* and 5 *assigned* as well as 4 *necessary* and 7 *assigned* operators is the next larger possible variation. Moreover, we state a system with higher costs but significant fall-out rate of 6 operators, consisting of 4 *necessary* and 10 *assigned* operators, which would raise the processing time of recovering key requests. Consequently, if we want to compare the results of all combinations we have to start the investigation with a minimum of 4 operators, because this is the maximum number of necessary operators in our example.

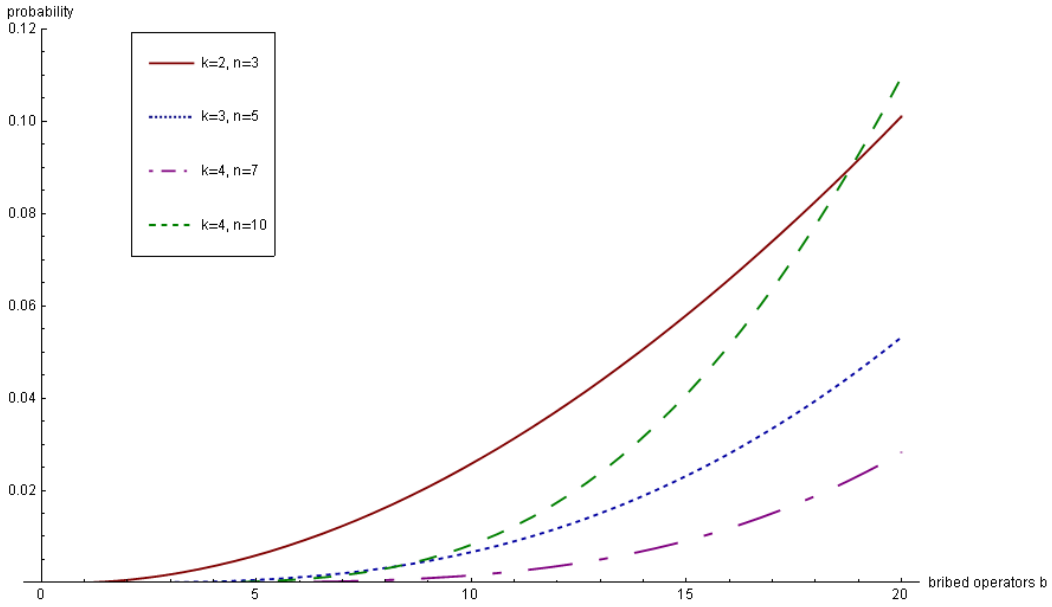

Figure 12: Different Combinations of Assigned and Necessary Operators for a Sample of 20 Bribed Operators

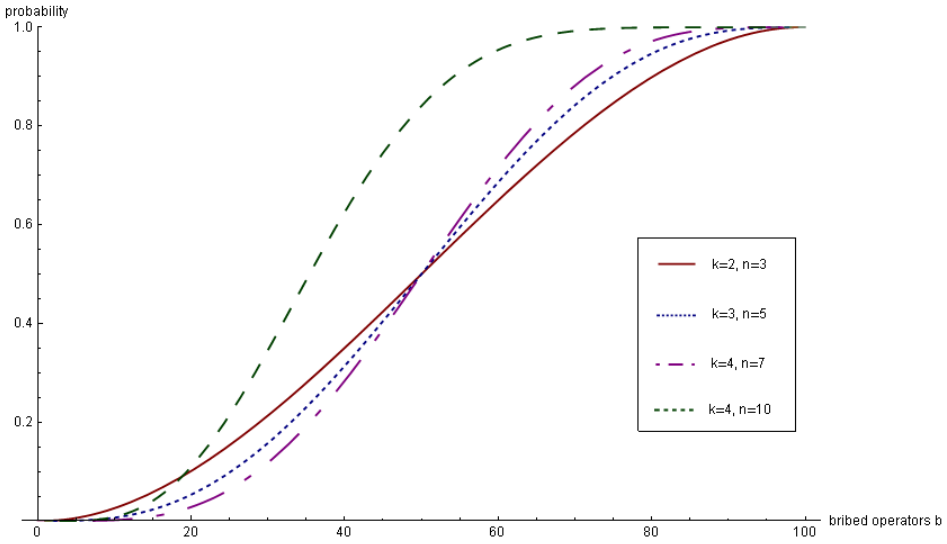

Figure 13: Different Combinations of Assigned and Necessary Operators for the Whole System

Figure 12, based on the results of Equation (4), shows that the probability for bribing 4 operators and a combination of $k = 2, n = 3$ tends towards a probability of 0.005, in other words 0.5 percent, whereas the percentages of all other constellations are still nearly zero. If an attacker is able to bribe $b = 10$ operators, the probability of compromising the privacy of a certain user in a system with $k = 2, n = 3$ operators is somewhat 2 percent, for $k = 3, n = 5$ as well as $k = 4, n = 10$

slightly more than 0.5 percent. The combination of 4 necessary and 7 operators still tends towards zero.

If in average 20 out of 100 operators are acting corruptly, it is the first time the approach representing the largest number of backup operators ($k = 4, n = 10$) gains the lowest security. Again, there is a significant difference between the previous mentioned approaches, which reach up to somewhat 10 percent, compared to 5 percent for $k = 3, n = 5$ and slightly more than 2 percent for $k = 4, n = 7$. Hence, the best security for a maximum of 20 percent of bribable operators can be achieved with 4 necessary and 7 assigned operators. Regarding the whole system (cf. Figure 13) consisting of 100 operators, the approach with the most backup operators performs worst from a security point of view, though, it can still be considered nearly as reliable as the cheapest variant.

If an attacker is able to bribe half of the operators within a system, all other approaches tend towards a probability of 50 percent for a successful attack. Consequently, the approach of 4 necessary and 7 assigned operators is the most secure combination in a system with 100 operators, bribing not more than about 50 percent.

Hence, in the following we provide a comparison of different system sizes for the combination of 4 necessary and 7 assigned operators.

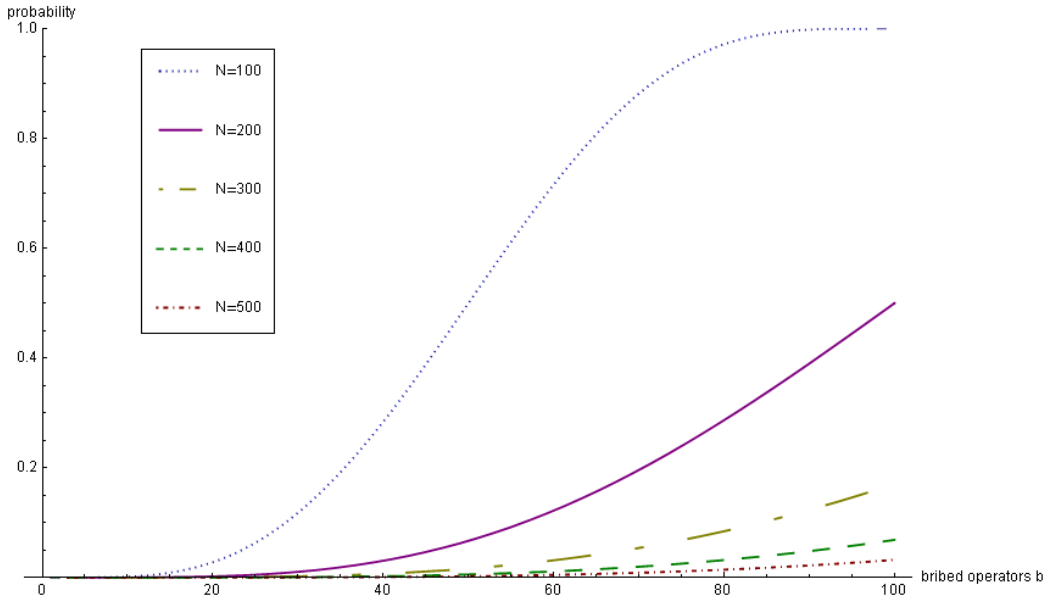

Figure 14: The Combination of 4 Necessary and 7 Assigned Operators Compared for Varying System Sizes

Figure 14 shows that for 20 operators the probability of bribing the minimum number of oper-

ators, necessary to rebuild a certain user's key, is converging to zero for all system sizes equal or greater 200 operators. If 50 operators are acting corruptly, the probabilities for system sizes with 300, 400 and 500 operators are still a few percent, whereas for a total of 200 operators it is slightly more than 5 percent. Regarding a number of 100 bribed operators, which represents one fifth for a system of 500, the probability is still significantly lower than 5 percent. This result can be compared to a system of 100 operators with 20, 200 operators with 40, 300 operators with 60 and 400 operators with 80 bribed operators. Hence, the security of the combination of 4 necessary and 7 assigned operators is also reliable for different system sizes.

We conclude that the maximum level of security respectively the lowest probability to bribe enough operators to find out a certain user's key may be gained with 7 assigned and 4 necessary operators. In this approach 3 backup operators exist, which provides a reliable fall-back mechanism as well.

Although this backup-mechanism assures a very high level of security, an increasing number of operators would result in high operational costs, especially if all operators are human beings. Thus we thought of a combination of humans with smart cards and hardware security modules (HSM) [103] which, if operated as trusted instances and separated in different places, could act on behalf of human operators and still provide a reliable system.

4.5.2 Backup Keystore with our Two-folded Approach

In this section we introduce our two-folded threshold variant regarding a secure backup of the users' inner private keys. Following Shamir, two parameters can be defined for sharing a secret, (i) the number of issued shares n and (ii) the amount of shares k that are necessary to re-establish the certain secret. The higher the number of issued shares compared to the number of shares that is needed to re-establish a shared secret, the higher the level of security, assuming the operators are randomly assigned, each holding one share of a certain secret.

Certainly, decrypting operations conducted by humans cause higher costs than performed by computers. To decrease the costs for establishing a backup keystore, we propose a combination of human operators $\mathcal{H} \subset \mathcal{O}$ and computers, which we call machine operators $\mathcal{M} \subset \mathcal{O}$. In this publication we state the conditions that both human and machine operators are required for recovering a secret, which is guaranteed by applying a two-folded variant of Shamir's secret sharing scheme.

In Figure 15, we show how a patient's inner private key $\widehat{K}_A^{-1}$ is divided amongst the human $\mathcal{H}$ and machine operators $\mathcal{M}$. Firstly, we divide the patient's inner private key into two parts, $\sigma_{\mathcal{H}}(\widehat{K}_A^{-1})$ for the human operators and $\sigma_{\mathcal{M}}(\widehat{K}_A^{-1})$ for the machine operators. Afterwards, the

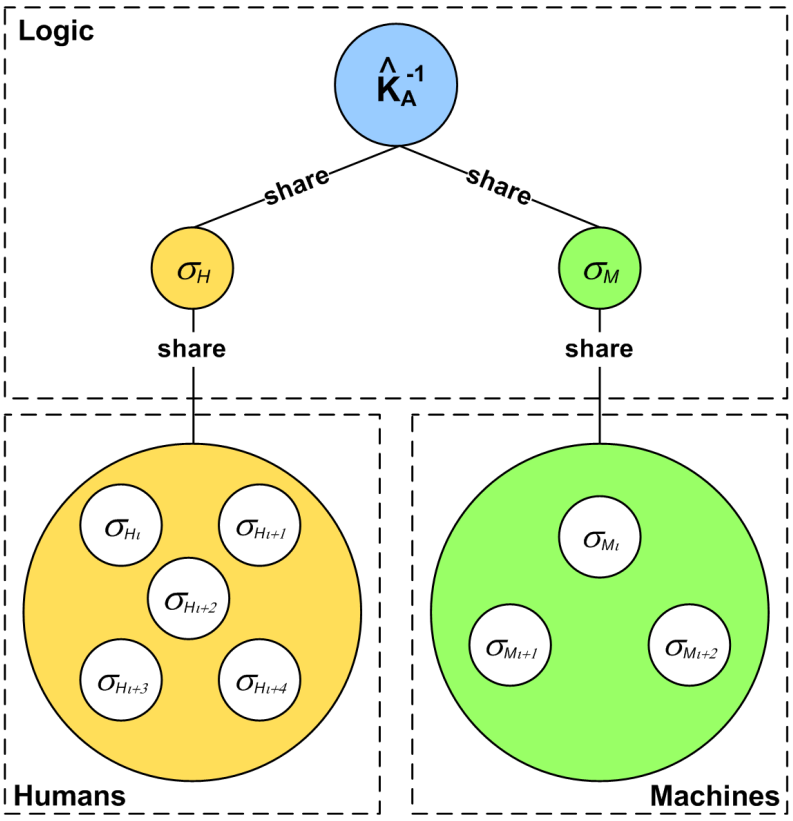

Figure 15: Two-folded Variant of Shamir's Threshold Scheme

threshold scheme is again applied to subdivide $\sigma_{\mathcal{H}}(\widehat{K}_A^{-1})$ into the number of n_H assigned human operators. Analogous $\sigma_{\mathcal{M}}(\widehat{K}_A^{-1})$ is distributed amongst n_M assigned machine operators. Finally, PIPE encrypts the shares σ_{H_ι} and σ_{M_ι} with a system's key and sends them to the particular assigned operators.

Using encryption helps to conceal the relation between an operator and a patient. If an operator has been successfully bribed, the attacker only gains access to her part of a user's shared secret, but not to the related parts of the other participants. The delta between the number of necessary operators k and assigned operators n serves as major availability constraint in our system, because a human operator may be ill or a machine operator could have a malfunction.

In the next section we present an example with real-life numbers. We show the costs for implementing and operating such a backup system as aforementioned.

4.5.3 Economical Aspects

The following example provides an overview of the costs, which we split in initial costs $C_{initial}$ (cf. Equation 5) and current costs $C_{current}$ (cf. Equation 10). The latter is based on Equation 7,

which is used for calculating the estimated key recovering requests per year and the Equations 8 and 9 for the numbers of human as well as machine operators. Moreover, we state the time to set-up the system, in other words the initial time $t_{initial}$ (cf. Equation 6). We define the necessary parameters for our obligations in Table 7.

abbr.	description
k_H	necessary human operators
k_M	necessary machine operators
n_H	assigned human operators
n_M	assigned machine operators
sc	smart card lifetime
p	percentage of lost smart cards per year
r	estimated requests per year
r_H^i	manageable requests of one human operator at set-up
r_M^i	manageable requests of one machine operator at set-up
r_H^c	manageable requests of one human operator per year
r_M^c	manageable requests of one machine operator per year
C_H	costs of one human operator including overhead
C_M	accumulated prime and maintenance costs
	of one machine operator per year

Table 7: Definition of Abbreviations Used in Economical Calculations

$$C_{initial} := U * \left(\frac{n_H * C_H}{r_H^i} + \frac{n_M * C_M}{r_M^i} \right) \tag{5}$$

$$t_{initial} := \begin{cases} \frac{n_H * U}{r_H^i} \\ \frac{n_M * U}{r_M^i} \end{cases} \tag{6}$$

$$r := \frac{U}{sc} + U * \frac{p}{100} \tag{7}$$

$$|\mathcal{H}| := \begin{cases} \frac{r * k_H}{r_H^c} & \text{if } \frac{r * k_H}{r_H^c} \geq n_H \\ n_H & \text{else} \end{cases} \tag{8}$$

$$|\mathcal{M}| := \begin{cases} \frac{r * k_M}{r_M^c} & \text{if } \frac{r * k_M}{r_M^c} \geq n_M \\ n_M & \text{else} \end{cases} \tag{9}$$

$$C_{current} := |\mathcal{H}| * C_H + |\mathcal{M}| * C_M \tag{10}$$

We assume an EHR system for 50 million users, which would depict the population of England [93]. A typical human operator with adequate education and experience would earn about Euro 36,000 [92] per year. The overhead costs, which will occur for example by working place expenses or equipment, result in 40 percent surplus. Hence, the total costs for one human operator C_H would be approximately Euro 50,000 per year. An average human works 200 days a year [83], which results in about 1,600 working hours on full-time employment. A human needs about 30 seconds to control a case of a lost smart card. Thus, she is able to contribute 192,000 requests r_H^c a year to recover inner private keys $\widehat{K}_A^{-1}$. Note that this number does not include the identification task and only refers to one of the necessary human operators $\mathcal{H}^k$.

As we already mentioned, it is possible to add machine operators M to the system to decrease the operational costs. The total costs C_M for these machines can be split into prime costs divided by lifetime and maintenance costs per year. We assume that the prime costs, the implementation costs and the running costs are Euro 10,000 each for an appropriate HSM with a lifetime of 10 years[13),14)].

This results in Euro 3,000 for a machine operator per year including all overhead costs. In our case a HSM is able to handle 360,000 operations per hour, which means that it could conduct 3,150,446,400 requests r_M^c a year in best case — calculated with an uptime of 99.9 percent. We assume that this number is equivalent at set-up time. During the initiation of the system, all assigned human operators have to encrypt the key shares $\sigma_{H_\iota}(\widehat{K}_A^{-1})$ and the machine operators the key shares $\sigma_{M_\iota}(\widehat{K}_A^{-1})$ of every participating user.

Opposite to that thesis, human operators are able to conduct more requests at set-up time compared to the yearly manageable requests because the encryption of the secret shares can be done in bulk. Therefore, these requests are only limited by the smart cards' runtime, which is not more than 1 second per operation. This leads to a total number of 5,760,000 manageable requests r_H^i on set-up for human operators. Regarding the smart card constraints, the typical lifespan is 5 years and we assume that the loss rate counts up to approximately 7 percent per year.

Shamir stated, that a minimum of $n = 2k - 1$ users, in our case operators, is required to re-

[13)] nCipher assumes a lifetime of 14 years and initial costs of Euro 12,700 for a nShield PCI 500, this device is able to handle 500 requests per second. Online: `http://www.ncipher.com`

[14)] Utimaco assumes a lifetime of 10+ years and initial costs of USD 9,600 for a Safeguard SecurityServer S10, this device is able to handle 100 request per seconds Online: `http://www.utimaco.us`

calculate a certain secret, which makes a "very robust key management scheme" [131]. A system with 5 assigned/3 necessary human and 3 assigned/2 necessary machine operators as well as the constraints defined above can be handled by 211 human operators and 3 machine operators. Nevertheless, we use 5 machine operators, which will only result in additional costs of Euro 6,000, because with 2 assigned and 3 necessary machine operators an attacker would know that every M has to hold a share of a certain user. Thus, in our example with $|\mathcal{H}| = 211$ and $|\mathcal{M}| = 5$, the initial costs are Euro 2,170,282 which is equal to Euro 0.043 per smart card. The current costs are Euro 10,565,000 per year or Euro 0.783 per worn-out, destroyed, stolen or lost smart card.

The initial set-up for our system comprises a two-folded process and lasts the time which is necessary to divide and distribute the users' key shares amongst the assigned human and machine operators. Hence, 250,000,000 shares for the human operators and 150,000,000 machine operator requests have to be handled by the total number of human and machine operators, if the shares will be randomly distributed. As already mentioned, the amount of manageable shares diverges between the current and initial manageable operations. The latter are used to allocate the occurring requests to the necessary time. Note, that in case the processes of the human and machine operators are started and run concurrently, the maximum of both time parameters results in the total initial time of ≈ 41.14 working days in our example.

In the following we elaborate on the security aspects of our example.

Security Obligations As we implemented the threshold scheme as two-folded process, an attacker is not able to reconstruct the user's inner private key until both sub secrets are successfully computed. Bribing $b_H \geq k_H$ human operators does not influence the probability of recovering the machine operators' sub secret $\sigma_{\mathcal{H}}$, too. Therefore, we are able to define these two events as statistically independent. Following the multiplication rule for independent events, the probability for their intersection, which means combining all necessary shares of a specific secret, is equivalent to the product of the single probabilities.

Table 8 provides a security overview of different combinations of bribed human and machine operators with the parameters of the example from the previous section, k_H=3, n_H=5 for $|\mathcal{H}| = 211$ human operators and k_M=2, n_M=3 for $|\mathcal{M}| = 5$ machine operators.

The first column shows the single probability of recovering the sub secret σ_H. This data can also be interpreted as security investigations on a single-folded process without the application of machine operators. In other words, an attacker has to guess only all necessary human operators for reconstructing an inner private key. In this scenario, the probability of bribing less than 5 assigned operators tends towards zero. If in average 10 operators, which corresponds to nearly

bribed	$P(\sigma_{\mathcal{H}})$	$P(\sigma_{\mathcal{M}})$	$P(\sigma_{\mathcal{H}} \cap \sigma_{\mathcal{M}})$
$b_H = 3, b_M = 2$	< 0,00001	0,3	< 0,00001
$b_H = 3, b_M = 3$	< 0,00001	0,7	< 0,00001
$b_H = 4, b_M = 2$	0,00003	0,3	< 0,00001
$b_H = 4, b_M = 3$	0,00003	0,7	0,00002
$b_H = 5, b_M = 2$	0,00006	0,3	0,00002
$b_H = 5, b_M = 3$	0,00006	0,7	0,00004
$b_H = 10, b_M = 2$	0,00074	0,3	0,00022
$b_H = 10, b_M = 3$	0,00074	0,7	0,00052
$b_H = 20, b_M = 2$	0,00651	0,3	0,00195
$b_H = 20, b_M = 3$	0,00651	0,7	0,00456
$b_H = 30, b_M = 2$	0,02144	0,3	0,00643
$b_H = 30, b_M = 3$	0,02144	0,7	0,01501

Table 8: Different Combinations of Bribed H and M Operators

5 percent of all 211 operators, act corruptly, the probability of compromising the privacy of a certain user still only amounts to less than 0.1 percent. Regarding a number of 30 bribed operators, the probability raises up to slightly more than 2 percent. Hence, even the application of a single-folded threshold scheme provides appropriate security.

As aforementioned, adding machine operators to the system with concurrent application of our two-folded approach helps to balance the operational costs and the security level of the system. The second column of Table 8 presents the probabilities of re-establishing the sub secret σ_M — 30 percent and 70 percent — which seems rather high, if taken out of the context that we applied a two-folded threshold scheme approach. However, reconsidering the above-mentioned constraints in our example implies that a successful attack is still decreased by adding only a few machine operators. In other words, if there were solely human operators, the probability of guessing at least all necessary shares for a certain user by bribing 20 human operators would amount to less than 0.7 percent, whereas applying our two-folded secret sharing scheme additionally reduces this probability to slightly less than 0.2 percent for 2 bribed machine operators respectively and approximately 0.5 percent for 3 bribed machine operators.

Depending on the available resources and time constraints, decision makers may decide if they want to use a single- or a two-folded backup approach to secure their users' keys. Anyway, we showed that misuse is already lowered significantly, if a threshold scheme is used instead of handing over the rights to access certain keys to single operators. Nevertheless, security cannot solely achieved with means of technology, but stakeholders should also bear organizational security in mind.

Summary In this section we introduced PIPE's architecture. We described the participating roles and components. Moreover, we discussed how pseudonymization can be used in order to setup a secure EHR solution.

As we increase the level of security by mainly giving authorizations by encryption techniques, the demand arises to establish a secure backup approach. — Regarding this keystore, we showed how efficiency, security and financial aspects can be balanced. This technique assures that all users still have access to their data in case of a lost or destroyed smart card. Thus, it provides a vital function of PIPE.

In the following we introduce the formal workflows of our system which are based on the discussed architecture. Please note, that all workflows have to be fully transparent to the end-users and operators to fulfill the security design principle of psychological acceptability (cf. Section 3.4.8). We used a formal language to clearly state the information flows between the actors. Thus, we are able to show where vulnerabilities may occur. With this methodology we can further investigate which information is transfered and what attributes need to be encrypted at which workflow step.

5 Formal Workflows

> "A workflow is the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules."
>
> *Fischer [47]*

In this section we firstly specify PIPE's security model which is based on the architecture of the previous section. Secondly, we introduce the used functions (cf. Section 5.1) as well as the administrative (cf. Section 5.2) and operational workflows (cf. Section 5.3) of PIPE. Then we show our ad-hoc authentication technique for the emergency access (cf. Section 5.4), which is also based on a novel pseudonymization approach. As conclusion we summarize the differences and benefits of PIPE in contrast to existing EHR approaches (cf. Section 5.5).

Our proposed system consists of users $\mathcal{U}$, which are mapped to the roles patient, relative, health care provider and operator. The patient (A), as the owner of her data, is in full control of her datasets. Every patient may give one or more relatives (B) the right for accessing all of her medical data. A health care provider (C) can be authorized to see or create a subset of the medical data by the patient. The operators (O), as we defined our administrative roles, share the secrets of the patients to provide a fall-back mechanism for lost, compromised or destroyed smart cards.

In addition we propagate the splitting of access rights within a user group instead of permitting a single health care provider with access to a certain anamnesis, diagnosis or treatment. This can be realized by the appliance of a threshold scheme. Following Shamir it is possible to establish a hierarchy between several actors by issuing more than one share to a chosen subset (cf. Section 3.3.4) [131]. In other words, if there is a minimum of k shares necessary to access a certain medical dataset, a general practitioner may receive more shares than a nurse. Moreover, this method may also be used for emergency data access.

Table 9 gives an overview of the keys and abbreviations used to describe our system. Note, that all private keys (where K stands for key) are identified as K^{-1}, for example the patient's inner private key will be named $\widehat{K}_A^{-1}$. All data is held persistent in the storage St, which represents the database as well as a secured keystore. In practice the logic L, and the storage St, which might be outsourced to a data processing center, form a trusted instance, because smart card management is handled there.

As shown in Figure 16, our system is based on a hull-architecture [107–114]. Every hull consists

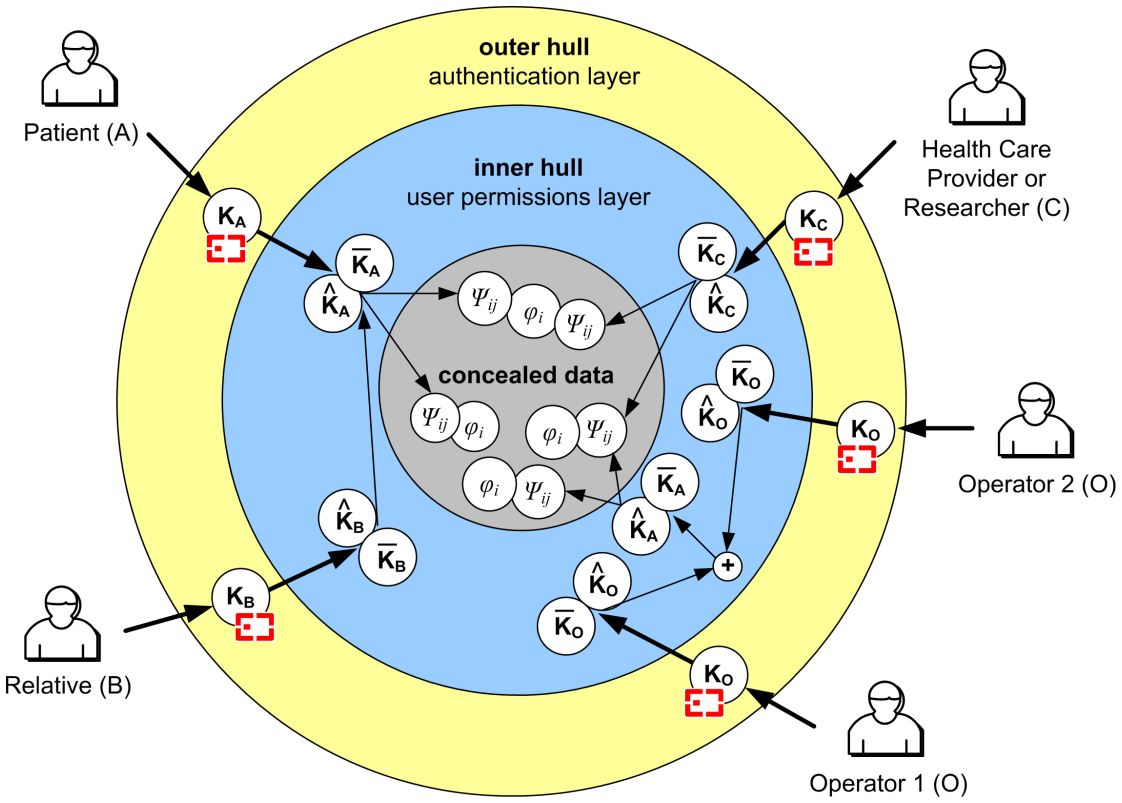

Figure 16: Layered Model Representing the Authorization Mechanism

	Patient	*Relative*	*HCP*	*Operator*	*Logic*
abbreviation	A	B	C	O	L
unique identifier	A_{id}	B_{id}	C_{id}	O_{id}	
(outer public key, private key)	(K_A, K_A^{-1})	(K_B, K_B^{-1})	(K_C, K_C^{-1})	(K_O, K_O^{-1})	
(inner public key, private key)	$(\widehat{K}_A, \widehat{K}_A^{-1})$	$(\widehat{K}_B, \widehat{K}_B^{-1})$	$(\widehat{K}_C, \widehat{K}_C^{-1})$	$(\widehat{K}_O, \widehat{K}_O^{-1})$	$(\widehat{K}_L, \widehat{K}_L^{-1})$
inner symmetric key	$\overline{K}_A$	$\overline{K}_B$	$\overline{K}_C$	$\overline{K}_O$	$\overline{K}_L$
key share	$\sigma_\kappa(K)$				
medical data	φ_i				
pseudonym	ψ_{i_j}				
tags	τ_v				
emergency access pseudonym	η_{A_i}		η_{C_i}		
emergency data pseudonym	ψ_{i_ϵ}				
emergency data	ϵ_A				

Table 9: Definition of PIPE's System Attributes

of one or more secrets (for example encrypted keys or hidden relations) which are only accessible with the unveiled secrets from the next outer hull. For instance, the patient's inner private key $\widehat{K}_A^{-1}$ in the inner hull — or user permissions layer of the patient A — is encrypted with the outer public key K_A on her smart card, which represents the outer hull or authentication layer. Moreover, a specific medical dataset φ_i, which is associated with a list of j pseudonyms ψ_{i_j},

can only be accessed with the knowledge of the related secret, which has been encrypted with the inner symmetric key $\overline{K}_A$. As the inner symmetric key has been preliminary encrypted with the inner public key, this encryption operation has to be reversed to gain access to this key in plaintext. In other words, if a patient wants to access her data, she firstly has to decrypt her inner private key $\widehat{K}_A^{-1}$, which is stored encrypted inside the system with the outer public key K_A of her smart card. Secondly, she is able to decrypt the inner symmetric key $\overline{K}_A$ with her inner private key. Afterwards she can use the inner symmetric key, which is now available for her in plaintext, to get access to the encrypted secrets in the most inner hull, the concealed data hull, by decrypting them. Consequently, to get access to the data, every user has to "peel the hulls".

In our system, secrets can be shared between users for authorization purposes. First of all, a patient can provide a relative with her inner private key $\widehat{K}_A^{-1}$, which will then be encrypted with the relative's inner public key $\widehat{K}_B$. Hence, the relative gets access to all data of the patient, until the inner private key is changed. In case, authorizations shall solely based on encryption instead of using a role-based access control model, an alternative for permitting a certain relative's access would be the same technique which we use for health care providers.

A health care provider can be authorized to access a subset of anamnesis, diagnosis or treatment datasets by sharing secrets, in our approach pseudonyms, of the concealed hull. A special case of a pseudonym, a so-called root pseudonym ψ_{i_0} exists for every dataset. This root pseudonym is only related with the patient and the medical data and no other user than the patient herself is able to delete this pseudonym. All other pseudonyms may be removed from the storage without authorized users permission. For example, if two health care providers are related to see a specific medical dataset, three pseudonyms exist, whereas both pseudonyms which are shared between a patient and a health care provider may be deleted without the particular health care providers notification. Moreover, if the patient decides to delete the anamnesis, treatment or diagnosis dataset, she is the only user, for whom it is possible to delete all pseudonyms. This assures that the patient is in full control of her data, and authorizing as well as revoking of all users is possible at any time.

The following equations summarize each actors access to her data.

Patient's access

$$\left\{\left\{\left\{\{\psi_{i_0} \mapsto \varphi_i\}_{\overline{K}_A}\right\}_{\widehat{K}_A}\right\}_{K_A}\right\} \tag{11}$$

Relative's access

$$\left\{\left\{\left\{\left\{\{\psi_{i_0} \mapsto \varphi_i\}_{\overline{K}_A}\right\}_{\widehat{K}_A}\right\}_{\widehat{K}_B}\right\}_{K_B}\right\} \tag{12}$$

HCP's access

$$\left\{\left\{\left\{\{\psi_{i_j} \mapsto \varphi_i\}_{\overline{K}_C}\right\}_{\widehat{K}_C}\right\}_{K_C}\right\} \tag{13}$$

Operators's access

$$\left\{\left\{\left\{\left\{\sigma_\kappa(\widehat{K}_A^{-1})\right\}_{\overline{K}_O}\right\}_{\widehat{K}_O}\right\}_{K_O}\right\} \tag{14}$$

For information exchange between two or more actors, we use the notation of the i^{th} workflow step: $Sender \rightarrow Receiver \rightarrow \ldots \rightarrow Receiver{:}\{Message\}$. If we apply a key as subscripted character, the message has been encrypted with this key. Every attribute which we state in a bracket, has been encrypted as a single attribute and is therefore also decryptable individually. This measurement is also necessary for querying encrypted texts in the database.

In the following, we state an example how the third message in a workflow between the patient and the logic, encapsulating the patient's identifier encrypted with the patient's inner symmetric key would look like.

Instance for a workflow step 3: $A \rightarrow L{:}\left\{\{A_{id}\}_{\overline{K}_A}\right\}$

In the next sections we introduce the necessary operations and constraints to conduct the workflows of our approach. Furthermore, we show how the authorizing and revoking of users works.

5.1 Used Functions

In this section we introduce the basic functions which we use throughout the administrative and operational workflows.

5.1.1 Calculate Hash

Due to performance reasons it is necessary, to use hash-values of data instead of applying for example a signature algorithm to a long text. We depict the function to hash some information in Equation 15.

$$f_{hash}(data) := \{hashed\ data\} \tag{15}$$

5.1.2 Sign Message

For integrity purposes and to support our mutual authentication protocol, we introduce the formal definition of our signing function in Equation 16. As stated in the previous section, in case of large data, we do not create the signature from the complete data, but rather use the signing function on the hash of this data.

$$f_{sign}(data, K^{-1}) := \{signed\ data\} \tag{16}$$

As stated in Equation 17, the receiver needs the transmitted data, the signature and the sender's public key to verify the validity of a signature.

$$f_{verify}(data, K, signature) := \left\{ \begin{array}{c} ok \\ errorcode \end{array} \right\} \tag{17}$$

5.1.3 Authenticate User

We sub sum the mutual authentication process in Equation 18.

$$f_{authenticate}(U_{id}) := \left\{ \begin{array}{cc} \left\{ \left\{ \widehat{K}_U^{-1} \right\}_{K_U}, \left\{ \left\{ \overline{K}_U \right\}_{\widehat{K}_U} \right\} \right\} & U_{id} \in St \\ errorcode & U_{id} \notin St \end{array} \right\} \tag{18}$$

The authorization workflow, which is a necessary pre-condition for any other workflow, consists of the following steps.

1: $U \rightarrow L$:$\left\{ \left\{ f_{sign}(U_{id}, K_U^{-1}) \right\}, \{U_{id}\} \right\}$

The user U authenticates against her smart card by entering her PIN. If the PIN matches, the user's outer private key is used to sign the user's identifier U_{id}. Then, the user sends the signed identifier and her plaintext identifier to the logic.

Necessary operations: sign user's identifier

2: $L \rightarrow St$:$\{U_{id}\}$

3: $St \rightarrow L$:$\{K_U\}$

The logic queries the storage with the user's identifier to receive the user's public key for further comparison. The logic trusts the user's certificate, because it trusts its own storage. As an alternative, an external TTP may be used, which stores the certificates on behalf of the system.

Necessary operations: one SQL select statement

4: $L \rightarrow St$:$\{U_{id}\}$

Thus, if the signature is also valid, the logic has unambiguously identified the user, because only the user is able to access the outer private key on her smart card. Therefore, the logic queries the storage for the encrypted inner keys of the specific user.

Necessary operations: verify signature of user's identifier

5: $St \rightarrow L$:$\left\{ \left\{ \widehat{K}_U^{-1} \right\}_{K_U}, \left\{ \overline{K}_U \right\}_{\widehat{K}_U} \right\}$

The storage replies with the encrypted inner private key $\widehat{K}_U^{-1}$ and the encrypted inner symmetric key $\overline{K}_U$.

Necessary operations: one SQL select statement

6: $L \rightarrow U$:$\left\{ \left\{ f_{sign}(U_{id}, \widehat{K}_L^{-1}) \right\}, \left\{ \widehat{K}_U^{-1} \right\}_{K_U}, \left\{ \overline{K}_U \right\}_{\widehat{K}_U} \right\}$

The logic also signs the user's identifier with her key inner private key $\widehat{K}_L^{-1}$ and forwards this ciphertext and the received encrypted keys to the user.

The user verifies the logic's signature of her identifier by the usage of the logic's public key $\widehat{K}_L$, which is integrated in the client software. This key is exchanged regularly over a secured channel. Alternatively the user may also query a TTP for the key to unambiguously identify her communication partner.

After confirming the identity of the logic, the user decrypts her inner private key $\widehat{K}_U^{-1}$ with her outer private key and subsequently her inner symmetric key with her inner private key.

Necessary operations: verify logic's signature of user's identifier, decrypt inner private key, de-

crypt inner symmetric key

With this workflow we assure the avoidance of man-in-the-middle attacks, because only the communication partners possess their particular private keys. Thus, only the logic and the specific user are able to calculate the necessary signatures. Moreover, from a security point of view, even if the client application has been successfully compromised, an attacker is not able to gain access to the pseudonymized datasets, because she would also need the outer private key to decrypt the encrypted inner private key.

The user's and logic's public keys can also be used to establish a secured channel, for example by TLS. Please note, that due to efficiency reasons the particular public keys are only used to exchange a symmetric key in TLS. The latter symmetric key is also changed regularly to enhance the communication security.

5.2 Administrative Workflows

In this section we present the workflows of how new actors can be added to the system and how the information of lost or destroyed smart cards can be recovered.

5.2.1 Adding an Actor to the System with One Operator Type

Every new actor in the system needs a smart card with her outer key pair stored on it, and a dataset in the identification database which includes the inner public key as well as encrypted versions of the inner symmetric and inner private key. Furthermore, the inner private key will be shared by $\mathcal{O}^n$ operators for backup reasons.

1: $O \rightarrow L$:$\{f_{authenticate}(O_{id}) =?\} \forall \mathcal{O}^n$

As we apply a threshold scheme on the user's inner private key within this workflow, a precondition for adding a new user is, that a minimum of $\mathcal{O}^n$ assigned operators have already been authenticated against the logic.

Necessary operations: mutual authentication for a minimum of $\mathcal{O}^n$ operators

2: $U \rightarrow O \rightarrow L$:$\{U\}$

After identification against one or more persons (for example by applying the four-eye-principle as an organizational security aspect), the user is able to send her personal information to the logic. The logic calculates a unique identifier U_{id} for the user by applying $f_{hash}(U)$.

Necessary operations: proof of user's identity, execute a hash-algorithm to compute the unique identifier U_{id}

3: $L \rightarrow St$:$\{\exists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ ?\}$

The logic looks up the storage to check if the identifier U_{id} is available. This verifies that the user does not exist in the system.

Necessary operations: conduct one SQL select statement

4: $St \rightarrow L$:$\left\{\nexists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ , \left\{\widehat{K}_U\right\}, \left\{\widehat{K}_U^{-1}\right\}, \left\{\overline{K}_U\right\}, \left\{K_U\right\}, \left\{K_U^{-1}\right\}\right\}$

After the storage replied with 'U_{id} unknown', the new identifier combined with the personal information is stored in the identification database. Furthermore, the logic receives an inner key pair, inner symmetric key and the user's outer key pair from the secured keystore and initiates the smart card production. The personalized smart card will hold the outer public key pair as well as the user's identifier and is secured by a randomly generated PIN.

Necessary operations: retrieve two asymmetric key-pairs and a symmetric key of secured keystore, initiate smart card production

5: $L \rightarrow St$:$\left\{\left\{\widehat{K}_U^{-1}\right\}_{K_U}, \left\{\overline{K}_U\right\}_{\widehat{K}_U}\right\}$

In the following, the logic encrypts the inner private key with the user's outer public key and the inner symmetric key with the inner public key and then forwards these ciphertexts to the storage.

Necessary operations: encrypt two keys and execute one SQL insert statement

6: $L \rightarrow O$:$\left\{\left\{\left\{\left\{\sigma_\kappa(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\widehat{K}_O}\right\} \forall\ \mathcal{O}^n$

Afterwards, the logic randomly selects $\mathcal{O}^n$ operators and applies the threshold scheme to divide

the user's inner private key into n secret shares σ_κ. All shared secrets will be first of all encrypted with the logic's inner symmetric key $\overline{K}_L$, subsequently encrypted with the inner public key $\widehat{K}_O$ of the particular operators and finally send to the operators. Moreover, the logic encrypts the user's identifier with the logic's inner symmetric key and subsequently conducts another encrypt operations on the gained ciphertexts with the particular operators key. Finally, the logic forwards these ciphertexts to the operators.

Necessary operations: apply threshold scheme, encrypt shares and user's identifier twice for $\mathcal{O}^n$ operators

$$7\text{: } O \rightarrow L \rightarrow St\text{:} \left\{ \left\{ \left\{ \left\{ \sigma_\kappa(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\overline{K}_O} \right\} \forall\ \mathcal{O}^n$$

The randomly assigned operators decrypt the received share and the user's identifier with their inner private keys $\widehat{K}_O^{-1}$. Then they encrypt both attributes again with their inner symmetric keys $\overline{K}_O$ and transmit these ciphertexts to the logic which forwards them to the storage.

Necessary operations: decrypt and encrypt the key shares and the user's identifier for $\mathcal{O}^n$ operators; $|\mathcal{O}^n|$ SQL insert statements to store the ciphertexts in the database

$$8\text{: } L \rightarrow St\text{:} \left\{ \left\{ \widehat{K}_U \right\}, \{K_U\} \right\}$$

$$9\text{: } L \rightarrow O \rightarrow U\text{:} \left\{ \left\{ K_U^{-1} \right\}, \{K_U\}, \{PIN\} \right\}$$

To assure secure posterior communication, the logic sends the user's inner and outer public keys to the storage and issues the new smart card with the outer key pair via the operator to the user. For access to the card, the user also receives a concealed letter, which holds the secret PIN code.

Necessary operations: execute one SQL update statement, finalize programming the smart card

In addition to the fact that the smart card programming and issuing component has to be a trusted instance, because of the key management component, we established a role-based access control model (cf. Section 4.3), which assures that no operator is allowed to see the ciphertext of the user's identifier encrypted with the logic's inner symmetric key $\{U_{id}\}_{\overline{K}_L}$ while processing (cf. workflow step 7). Otherwise it would be possible for the operators to write down and compare the encrypted identifier with other operators. This could lead to frauds against the system.

5.2.2 Adding an Actor to the System with our Two-folded Approach

In contrast to the previous section, we now introduce the workflow of adding a new actor, whereas the backup keystore is shared between two groups of operators. This is a measurement to decrease the costs. Thus, the inner private key will be shared by $\mathcal{H}^n$ assigned human operators and $\mathcal{M}^n$ assigned machine operators.

1a: $H \rightarrow L$:$\{f_{authenticate}(H_{id}) =?\}\, \forall \mathcal{H}^n$

1b: $M \rightarrow L$:$\{f_{authenticate}(M_{id}) =?\}\, \forall \mathcal{M}^n$

As our two-folded approach consists of two different groups of operators, a minimum of $\mathcal{H}^n$ assigned human operators and $\mathcal{H}^n$ assigned machine operators have already been authenticated against the logic.

Necessary operations: mutual authentication for a minimum of $\mathcal{O}^n = \mathcal{H}^n \cap \mathcal{M}^n$ operators

2: $U \rightarrow H \rightarrow L$:$\{U\}$

After identification against one or more human operators, the user is able to send her personal data to the logic. The logic computes a unique identifier U_{id} for the user by applying $f_{hash}(U)$.

Necessary operations: proof of user's identity, execute a hash-algorithm to compute the unique identifier U_{id}

3: $L \rightarrow St$:$\{\exists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ ?\}$

The logic looks up the storage to check if U_{id} already exists and hence verifies that the user has not been added to the system yet.

Necessary operations: conduct one SQL select statement

4: $St \rightarrow L$:$\left\{\nexists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ , \left\{\widehat{K}_U\right\}, \left\{\widehat{K}_U^{-1}\right\}, \left\{\overline{K}_U\right\}, \{K_U\}, \left\{K_U^{-1}\right\}\right\}$

After the storage replied with 'U_{id} unknown', the new identifier combined with the personal data is added to the identification database. Moreover, the logic retrieves an inner key pair, an inner symmetric key and the user's outer key pair from the secured keystore and starts the smart card

production off. The personalized smart card will hold the user's identifier and the generated outer public key pair. It is secured by a randomly generated PIN.

Necessary operations: retrieve two asymmetric key-pairs and a symmetric key of secured key-store, initiate smart card production

5: $L \rightarrow St: \left\{ \left\{ \widehat{K}_U^{-1} \right\}_{K_U}, \{ \overline{K}_U \}_{\widehat{K}_U} \right\}$

In the next step, the logic conducts the necessary encrypting operations and subsequently sends the inner private key, encrypted with the user's outer public key, as well as the inner symmetric key encrypted, with the inner public key, to the storage.

Necessary operations: encrypt two keys and execute one SQL insert statement

6a: $L \rightarrow H: \left\{ \left\{ \left\{ \left\{ \sigma_{\mathcal{H}_\iota}(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\widehat{K}_H} \right\} \forall\ \mathcal{H}^n$

6b: $L \rightarrow M: \left\{ \left\{ \left\{ \left\{ \sigma_{\mathcal{M}_\iota}(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\widehat{K}_M} \right\} \forall\ \mathcal{M}^n$

The logic randomly selects $\mathcal{H}^n$ human operators as well as $\mathcal{M}^n$ machine operators and uses the threshold scheme to divide the user's inner private key into the secret shares $\sigma_{\mathcal{H}}$ and $\sigma_{\mathcal{M}}$. All shares will be first of all encrypted with the logic's inner symmetric key $\overline{K}_L$, subsequently encrypted with the inner public key $\widehat{K}_O$ of the particular operators and finally send to the operators. Moreover, the logic encrypts the user's id with the logic's inner symmetric key as well as the particular operators key and transfers these ciphertexts to the operators too.

Necessary operations: apply threshold scheme, encrypt shares and user's identifier twice for $\mathcal{O}^n$ operators

7a: $H \rightarrow L \rightarrow St: \left\{ \left\{ \left\{ \left\{ \sigma_{\mathcal{H}}(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\overline{K}_H} \right\} \forall\ \mathcal{H}^n$

7b: $M \rightarrow L \rightarrow St: \left\{ \left\{ \left\{ \left\{ \sigma_{\mathcal{M}}(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\overline{K}_M} \right\} \forall\ \mathcal{M}^n$

The assigned human operators and machine operators decrypt the received ciphertexts with their inner private keys $\widehat{K}_O^{-1}$. They then encrypt both attributes again with their inner symmetric keys

$\overline{K}_O$ and send this secured information to the logic which forwards it to the storage.

Necessary operations: decrypt and encrypt the key shares and the user's identifier for $\mathcal{H}^n$ human operators and $\mathcal{M}^n$ machine operators; $|\mathcal{O}^n|$ SQL insert statements to store the ciphertexts in the database

8: $L \rightarrow St$:$\left\{\left\{\widehat{K}_U\right\}, \{K_U\}\right\}$

9: $L \rightarrow H \rightarrow U$:$\left\{\left\{K_U^{-1}\right\}, \{K_U\}, \{PIN\}\right\}$

The logic transfers the user's inner and outer public keys for posterior communication purposes to the storage and issues the new smart card with the outer key pair to a human operator. Finally, she hands out the newly produced smart card and the concealed PIN code to user.

Necessary operations: execute one SQL update statement, finalize programming the smart card

Due to the fact that we split the user's inner private key firstly into two shares and subsequently calculate the individuals operators's shares, we may define different numbers of assigned and necessary operators for each group of participating operators. As machine operators are much cheaper than their human equivalent, this is a cost-lowering initiative. Moreover, machine operators have faster reaction times. Nevertheless, both groups of operators are necessary to combine the shared secret.

5.2.3 Recovering a Lost Key by One Operator Type

In this section we elaborate on the recovery of a user's lost or destroyed smart card.

1: $O \rightarrow L$:$\{f_{authenticate}(O_{id}) = ?\}\, \forall \mathcal{O}^k$

Due to the constraints of a threshold scheme, we need a minimum number of $\mathcal{O}^k$ operators to re-establish the user's inner private key. Thus, the minimal number of necessary operators O^k need to be logged in for successful processing.

Necessary operations: mutual authentication for a minimum of $\mathcal{O}^k$ operators

2: $U \rightarrow O \rightarrow L \rightarrow St$:$\{\exists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ ?\}$

To replace a user's lost smart card, a new smart card has to be issued, which also allows access to the user's inner private key. To prevent frauds the user firstly identifies against an operator. As this operator only initiates the recovering process, it is not necessary that this particular operator has to hold a part of this user's inner private key. In fact, she just sends a message to the logic which indicates the start of the recovering process. Upon receipt, the logic checks via the storage if the user exists in the system.

Necessary operations: proof of user's identity, execute a hash algorithm to compute the user's unique identifier

3: $St \rightarrow L$:$\{\exists\ U \in \mathcal{U} : f_{hash}(U) = U_{id}\}$

The storage confirms the user's existence in the storage.

Necessary operations: conduct one SQL select statement

4: $L \rightarrow O$:$\left\{\{U_{id}\}_{\overline{K}_L}\right\} \forall\ \mathcal{O}$

Afterwards, the central logic broadcasts a message to all operators $\mathcal{O}$ with an encrypted version of the user's identifier U_{id} because as mentioned in the previous section, the logic's inner symmetric key $\overline{K}_L$ has been used to envelope the identifier first.

Necessary operations: encrypt user's identifier

5: $O \rightarrow L \rightarrow St$:$\left\{\left\{\{U_{id}\}_{\overline{K}_L}\right\}_{\overline{K}_O}\right\} \forall\ \mathcal{O}$

Upon receipt, all operators query their backup keystore via the central logic by encrypting these ciphertexts with the particular operator's inner symmetric key $\overline{K}_O$. With these messages the logic is able to find out which operator possesses a user's key share.

Necessary operations: encrypt shares by $|\mathcal{O}|$ operators

6: $St \rightarrow L \rightarrow O$:$\left\{\left\{\left\{\sigma_\kappa(\widehat{K}_U^{-1})\right\}_{\overline{K}_L}\right\}_{\overline{K}_O}\right\} \forall\ \mathcal{O}^n$

After querying the double encrypted ciphertexts against the storage, the logic receives the associated double encrypted key shares and forwards them to the assigned operators.

Necessary operations: a maximum of $|\mathcal{O}|$ SQL select statements

7: $O \rightarrow L$:$\left\{\left\{\sigma_\kappa(\widehat{K}_U^{-1})\right\}_{\overline{K}_L}\right\} \forall\ \mathcal{O}^k$

The next step is that all necessary operators decrypt their particular shared secrets with their inner symmetric key $\overline{K}_O$ and to transmit them to the logic. The logic is now able to decrypt these shares with its key K_L and consequently to combine the parts. As soon as the logic receives the shares from a minimum number of k necessary operators, the user's inner private key can be re-calculated with the appliance of Shamir's threshold scheme [131].

Necessary operations: decrypt a maximum of $|\mathcal{O}^k|$ key shares, apply threshold scheme

8: $L \rightarrow St$:$\{U_{id}\}$

Afterwards, the logic asks the storage for a new outer key pair.

Necessary operations: generate new asymmetric key pair

9: $St \rightarrow L$:$\left\{\left\{K_{U'}^{-1}\right\}, \{K_{U'}\}\right\}$

10: $L \rightarrow St$:$\left\{\left\{\widehat{K}_U^{-1}\right\}_{K_{U'}}\right\}$

The logic retrieves a new outer key pair $(K_{U'}, K_{U'}^{-1})$ from the storage which will replace the outer keys (K_U, K_U^{-1}) of the lost smart card. The logic uses the new outer public key to encrypt the user's inner private key which has been re-constructed with the threshold scheme. The logic saves this ciphertext in the storage and initiates the smart card production. Moreover, the logic informs the storage to update the user's outer public key in the database.

To avoid replay-attacks the storage moreover deletes the operator shares and their relations to the user.

Necessary operations: encrypt user's inner private key, one SQL update statement, O'^n SQL delete statements

11: $L \rightarrow O$:$\left\{\left\{\left\{\left\{\sigma_\kappa(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\widehat{K}_O}\right\} \forall\ \mathcal{O}'^n$

Subsequently, the logic randomly chooses $\mathcal{O}^n$ assigned operators and uses the threshold scheme to divide the user's inner private key into n shares. Once more, all shares will be double-enveloped. Firstly, the logic applies its inner symmetric key $\overline{K}_L$ and secondly, encrypts the gained ciphertexts with the certain inner public keys $\widehat{K}_O$ of the selected operators. These encrypted secret shares will then be transmitted to the operators. Moreover, the logic applies the same encryption procedures to the user's id U_{id} and transfers this ciphertext to the operators too.

Necessary operations: apply threshold scheme, encrypt shares and user's identifier twice for $\mathcal{O}'^n$ operators

$$12\text{: } O \rightarrow L \rightarrow St\text{:} \left\{ \left\{ \left\{ \left\{ \sigma_\kappa(\widehat{K}_U^{-1}) \right\}, \{U_{id}\} \right\}_{\overline{K}_L} \right\}_{\overline{K}_O} \right\} \forall\ \mathcal{O}'^n$$

Upon receipt, the assigned operators decrypt their particular shares and the user's identifier with their inner private keys $\widehat{K}_O^{-1}$. Then they encrypt both attributes again with their inner symmetric keys $\overline{K}_O$ and return these ciphertexts to the logic which saves them in the storage.

Necessary operations: decrypt and encrypt the key shares and the user's identifier for $\mathcal{O}'^n$ operators; $|\mathcal{O}'^n|$ SQL insert statements to store the ciphertexts in the database

$$13\text{: } L \rightarrow O \rightarrow U\text{:} \left\{ \left\{ K_{U'}^{-1} \right\}, \{K_{U'}\}, \{PIN\} \right\}$$

Afterwards, the logic finalizes the smart card production and delivers the smart card to the operator. This operator hands the smart card and the concealed new PIN code out to the user.

Necessary operations: finalize smart card production

With this workflow, we provide the recovery scenario for lost or destroyed smart cards. We confirm that after a strong identification task the user's smart card can be replaced. This scenario may also be used to quickly lock compromised smart cards.

To avoid replay attacks by the operators, we further propagate the usage of version-numbers together with the key shares. In that case, only encrypted key-shares and associated encrypted user identifiers with the highest version-number will be retrieved by the logic and forwarded to the operators. Moreover, additional seeding of the ciphertext may be achieved by regularly changing logic keys.

5.2.4 Recovering a Lost Key with our Two-folded Approach

This workflow is the continuation of the re-coverage process of the previous section. In this section we will discuss our two-folded threshold scheme approach, which we introduced to decrease the costs.

Two types of operators exist, human operators H and machine operators M. For both groups a subgroup of assigned and a subgroup of necessary operators exist. A minimum of $|H^k|$ and $|M^k|$ operators have to work together to re-calculate the user's secret.

1a: $H \rightarrow L$:$\{f_{authenticate}(H_{id}) =?\}\, \forall \mathcal{H}^k$

1b: $M \rightarrow L$:$\{f_{authenticate}(M_{id}) =?\}\, \forall \mathcal{M}^k$

We need a minimum number of $\mathcal{H}^k$ human operators and $\mathcal{M}^k$ machine operators to re-establish the user's inner private key. Thus, the minimum number of these necessary operators needs to be logged in for the successful appliance of the threshold scheme.

Necessary operations: mutual authentication for a minimum of $\mathcal{H}^k$ and $\mathcal{M}^k$ operators

2: $U \rightarrow H \rightarrow L \rightarrow St$:$\{\exists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\ ?\}$

In order to rebuild a lost smart card with access to the user's inner private key, the user identifies against an human operator. It is not necessary that this operator has to hold a part of this user's inner private key. In fact she just initiates the recovering process by sending a message to the logic. The logic itself checks via the storage, if the user exists in the system.

Necessary operations: verify user's identity, execute a hash algorithm to compute the user's unique identifier

3: $St \rightarrow L$:$\{\exists\, U \in \mathcal{U} : f_{hash}(U) = U_{id}\}$

If the user exists, the storage confirms that fact.

Necessary operations: one SQL select statement

4: $L \rightarrow O$:$\left\{\{U_{id}\}_{\overline{K}_L}\right\} \forall\, \mathcal{O}$

A message is generated by the central logic and sent to all operators $\mathcal{O}$ with an encrypted version of the user's identifier U_{id}. As mentioned in the previous section, to hide the user's identifier U_{id} the logic's inner symmetric key $\overline{K}_L$ is applied.

Necessary operations: encrypt user's identifier

5: $O \rightarrow L \rightarrow St$:$\Big\{\{\{U_{id}\}_{\overline{K}_L}\}_{\overline{K}_O}\Big\} \forall\ \mathcal{O}$

If an operator receives the message, she looks up her backup keystore via the central logic by encrypting the ciphertext with her inner symmetric key $\overline{K}_O$. The central logic is able to find out if an operator possesses a user's key share with this message.

Necessary operations: encrypt shares by $|\mathcal{O}|$ operators

6a: $St \rightarrow L \rightarrow M$:$\Bigg\{\Big\{\Big\{\sigma_{\mathcal{M}_\iota}(\widehat{K}_U^{-1})\Big\}_{\overline{K}_L}\Big\}_{\overline{K}_M}\Bigg\} \forall\ \mathcal{M}^n$

6b: $St \rightarrow L \rightarrow H$:$\Bigg\{\Big\{\Big\{\sigma_{\mathcal{H}_\iota}(\widehat{K}_U^{-1})\Big\}_{\overline{K}_L}\Big\}_{\overline{K}_H}\Bigg\} \forall\ \mathcal{H}^n$

The logic queries the storage with the double encrypted identifiers. The storage replies with the associated double encrypted key shares and forwards them to the assigned human and machine operators.

Necessary operations: a maximum of $|\mathcal{H}| + |\mathcal{M}|$ SQL select statements

7a: $H \rightarrow L$:$\Big\{\Big\{\sigma_{\mathcal{H}_\iota}(\widehat{K}_U^{-1})\Big\}_{\overline{K}_L}\Big\} \forall\ \mathcal{H}^k$

7b: $M \rightarrow L$:$\Big\{\Big\{\sigma_{\mathcal{M}_\iota}(\widehat{K}_U^{-1})\Big\}_{\overline{K}_L}\Big\} \forall\ \mathcal{M}^k$

In the following step, the assigned human operators decrypt their particular shared secrets with their inner symmetric key $\overline{K}_H$ and transmit them to the logic. The machine operators conduct the mirrored operation with their shared secrets and their inner symmetric keys $\overline{K}_M$. The logic is now able to decrypt these shares with its key $\overline{K}_L$ and consequently to combine the human shared secrets $\sigma_{\mathcal{H}_\iota}$ as well as the machine shared secrets $\sigma_{\mathcal{M}_\iota}$. This leads to both sub secrets, which are necessary to re-calculate the user's inner private key. In other words, the logic needs

the shares from a minimum number of H^k and M^k necessary operators for a successful appliance of Shamir's threshold scheme [131].

Necessary operations: decrypt a maximum of $|\mathcal{H}^k| + |\mathcal{M}^k|$ key shares, apply threshold scheme

8: $L \rightarrow St$:$\{U_{id}\}$

Afterwards, the logic queries the storage for a new outer key pair.

Necessary operations: generate new asymmetric key pair

9: $St \rightarrow L$:$\left\{\left\{K_U^{-1}\right\}, \{K_{U'}\}\right\}$

10: $L \rightarrow St$:$\left\{\left\{\widehat{K}_{U'}^{-1}\right\}_{K_{U'}}\right\}$

The logic retrieves the new outer key pair $(K_{U'}, K_{U'}^{-1})$ from the storage. This key pair replaces the lost outer keys (K_U, K_U^{-1}) of the smart card. The logic encrypts the user's inner private key with the new outer public key $K_{U'}$ and saves this ciphertext in the storage. Additionally, a new smart card is produced by the logic. Finally, the storage deletes the operator shares and their relations to the user to avoid replay-attacks and updates the user's outer public key.

Necessary operations: encrypt user's inner private key, one SQL update statement, O^n SQL delete statements

11a: $L \rightarrow H$:$\left\{\left\{\left\{\left\{\sigma_{\mathcal{H}_\iota}(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\widehat{K}_H}\right\} \forall\ \mathcal{H}'^n$

11b: $L \rightarrow M$:$\left\{\left\{\left\{\left\{\sigma_{\mathcal{M}_\iota}(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\widehat{K}_M}\right\} \forall\ \mathcal{M}'^n$

Afterwards, the logic applies the threshold scheme again and divides the user's inner private key into two shares, one for the human $\sigma_{\mathcal{H}}(\widehat{K}_U^{-1})$ and one for the machine operators $\sigma_{\mathcal{M}}(\widehat{K}_U^{-1})$.

After generating the individual operators' shares, the logic uses its key K_L to encrypt this secret. Then it conducts an encryption with the certain inner public keys $\widehat{K}_H$ of the selected human or $\widehat{K}_M$ of the selected machine operators on the gained ciphertexts. These encrypted secret shares will then be transmitted to other randomly assigned operators $\mathcal{H}'^n$ and $\mathcal{M}'^n$. Furthermore, the

logic uses the same encryption operations on the user's id U_{id} and transfers this ciphertext to the operators, too.

Necessary operations: apply threshold scheme, encrypt shares and user's identifier twice for $|\mathcal{H}'^n| + |\mathcal{M}'^n|$ operators

12a: $H \rightarrow L \rightarrow St$:$\left\{\left\{\left\{\left\{\sigma_{\mathcal{H}_\iota}(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\overline{K}_H}\right\} \forall\ \mathcal{H}'^n$

12b: $M \rightarrow L \rightarrow St$:$\left\{\left\{\left\{\left\{\sigma_{\mathcal{M}_\iota}(\widehat{K}_U^{-1})\right\}, \{U_{id}\}\right\}_{\overline{K}_L}\right\}_{\overline{K}_M}\right\} \forall\ \mathcal{M}'^n$

The assigned human and machine operators decrypt their received encrypted shares and user's identifier. They use their particular inner private keys $\widehat{K}_H^{-1}$ / $\widehat{K}_M^{-1}$ to decrypt the information. Subsequently, they again encrypt both attributes with their inner symmetric keys $\overline{K}_M$ / $\overline{K}_H$ and return these ciphertexts to the logic which saves them in the storage.

Necessary operations: decrypt and encrypt the key shares and the user's identifier for $|\mathcal{H}'^n|+|\mathcal{M}^n|$ operators; $|\mathcal{H}^n| + |\mathcal{M}^n|$ SQL insert statements to store the ciphertexts in the database

13: $L \rightarrow H \rightarrow U$:$\left\{\left\{K_{U'}^{-1}\right\}, \{K_{U'}\}, \{PIN\}\right\}$

Afterwards, the logic personalizes the smart card. The operator gives the smart card together with the concealed new PIN code to the user.

Necessary operations: finalize smart card production

This workflow shows how the access to the backup keys and their re-establishing works. Moreover, with this technique we assure that a user's old smart card is not usable any more.

5.2.5 Security Obligations

Form a security point of view, we have to observe the aforementioned workflows regarding the security attributes: Confidentiality, integrity, availability and non-repudiation.

As all workflows start with a mutual authentication process and the establishment of a secure channel by an encrypted protocol like TLS, the confidentiality of all sensitive data is assured.

Moreover, the logic trusts persons who inherit the role operator and vice versa. Therefore, any operator will encrypt and decrypt all messages as demanded by the workflow if these messages come from the logic. The logic also accepts all ciphertexts from all operators. If this trust-circle does not exist, there would be no possibility to assure integrity because as soon as the operators encrypted a certain text, which comes from the logic, only the particular operators know the content. Nevertheless, the role "operator" is important because it assures the security design principle of separation of duties.

The availability, which in our case means the assurance of business continuity of these administrative workflows, is a parameter which can be adopted to the system users' needs. The difference between assigned and necessary operators, which can be seen as backup operators, influences the processing time, costs and security. As already discussed in Section 4.5, the usage of more backup operators increases the time for adding actors to the system, but also increases the availability of recovering lost keys. Moreover, as we showed, the security is lowered as well, while the costs rise.

Non-repudiation can be assured by implementing a logging functionality. If all actions undertaken are stored and observed regularly, frauds may be barred or at least the risk and the frequency of their occurrence would be lowered. Moreover, signatures could also be used for non-repudiation, but in that case it would also be necessary to conceal the relation between the signature and signer, because otherwise an attacker, who gains access to the database may find out which operators share a secret for a certain user.

5.3 Operational Workflows

In this section we introduce the operational workflows of the system. We show how actors may add and share data. Furthermore, we continue the discussion about different roles and their permissions in PIPE.

5.3.1 Authorizing a User

A relation between two users is set up by exchanging their ids and mutually encrypting them with their inner symmetric keys. Moreover, any user may additionally share her inner private key, or in other words, access to all of her sensitive data with another user.

In the following we present the setup of an relation between a patient and a relative, which is the role with full access to another person's data, as a practical example.

1a: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

1b: $B \rightarrow L$:$\{f_{authenticate}(B_{id}) =?\}$

The patient and the relative log onto the system. After that, both have established a secure channel and are equipped with their inner keys.

Necessary operations: mutual authentication for both users

2a: $A \rightarrow L \rightarrow St$:$\{A_{id}\}$

2b: $B \rightarrow L \rightarrow St$:$\{B_{id}\}$

Afterwards, the patient initiates the exchange process by sending her identifier to the logic. The relative also sends her identifier to the logic. The logic forwards both identifiers to the storage.

Necessary operations: -

3a: $St \rightarrow L$:$\left\{\widehat{K}_B\right\}$

3b: $St \rightarrow L$:$\left\{\widehat{K}_A\right\}$

The storage replies with the particular inner public keys of the communication partners.

Necessary operations: two SQL select statements

4a: $L \rightarrow B$:$\left\{\{A_{id}\}_{\widehat{K}_B}\right\}$

Firstly, the logic encrypts the patients's identifier A_{id} with the relative's inner public key $\widehat{K}_B$ and transfers this ciphertext to the relative.

Necessary operations: one encrypt operation

4b: $L \rightarrow A$:$\left\{\{B_{id}\}_{\widehat{K}_A}\right\}$

Secondly, the logic conducts the opposite operation with the relative's identifier B_{id} and the patient's inner public key $\widehat{K}_A$ and sends this ciphertext to the patient.

Necessary operations: one encrypt operation

5a: $B \rightarrow L \rightarrow St{:}\left\{\left\{\{A_{id}\}, \{B_{id}\}\right\}_{\overline{K}_B}\right\}$

As the patient's identifier is encrypted, the relative needs to decrypt A_{id} with her inner private key $\widehat{K}_B^{-1}$ before she is able to apply her inner symmetric key $\overline{K}_B$. Moreover, she also encrypts her own identifier B_{id} with her inner symmetric key and transfers both attributes to the logic, which forwards it to the storage.

Necessary operations: decrypt and encrypt identifier, encrypt own identifier, one SQL insert statement

5b: $A \rightarrow L \rightarrow St{:}\left\{\left\{\{B_{id}\}, \{A_{id}\}\right\}_{\overline{K}_A}\right\}$

Compared to step 5a the patient conducts the mirrored operations. The patient's identifier will be decrypted with the patient's inner private key and afterwards encrypted again with the patient's inner symmetric key. The patient also encrypts her identifier and sends both to the logic which inserts the ciphertexts via the storage in the database.

Necessary operations: decrypt and encrypt identifier, encrypt own identifier, one SQL update statement

6: $A \rightarrow L \rightarrow St{:}\left\{\left\{\widehat{K}_A^{-1}\right\}_{\widehat{K}_B}\right\}$

Afterwards, the patients encrypts her inner private key with the relative's inner public key. She sends the ciphertext to the logic which forwards it to the storage. As a consequence the relative would gain the same rights as the patient if not controlled by a role-based access control model (cf. Section 4.3). As alternative a certain relative could only be authorized for a subset of the patient's data. We discuss this workflow in Section 5.3.7.

Necessary operations: encrypt inner private key, one SQL insert statement

Authorizing of the relation between a health care provider C and a patient A is similar to the workflow between a relative and a patient. In contrast, the patient does not provide the health care provider via the logic with her inner private key. Nevertheless, our role-based-access control model only allows adding of new data, if the users have setup a relation between them (cf. Section

4.3).

As previously stated, from a security point of view, every user only trusts the logic and the secured channel between herself and the logic. Thus, in this workflow, the logic retrieves the communication partner's inner public key from the storage via a query with the opposites identifier. Hence, an identifier may be spoofed, but an attacker could not use the data because she does not possess the matching inner private key for decrypting purposes. This technique not only assures confidentiality, but also integrity, because firstly the channels themselves are secured and secondly all transferred information is encrypted and cannot be changed during processing.

Moreover, as a matter of fact, private keys are only accessible for each particular user. This fact grants non-repudiation for this workflow. Nevertheless, we once more encourage the usage of log-files or concealed signatures. Regarding the assurance of the last security attribute, availability, as long as the logic is not compromised with a denial-of-service attack it is also guaranteed.

5.3.2 Revoking a User

As we already mentioned, we established a role-based access control model which controls if a user is allowed to add datasets for another one. For instance a health care provider may be authorized to query a subset of the patient's medical dataset, but she may only add additional entries for this certain patient with prior authorization. To revoke this user-related authorization, the entry in the storage has to be deleted. We present the workflow for revoking a health care provider by a patient as an example.

1: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

The patient authenticates against the system and establishes a secured channel.

Necessary operations: mutual authentication

2: $A \rightarrow L \rightarrow St$:$\left\{\{A_{id}\}_{\overline{K}_A}\right\}$

The patient encrypts her identifier A_{id} and sends it to the storage via the logic to look up the user-relations table.

Necessary operations: encrypt patient's identifier

3: $St \rightarrow L \rightarrow A$:$\left\{\{\{B_{id}\}, \{C_{id}\}\}_{\overline{K}_A}\right\}$

The storage replies with the set of users, which are assigned to the patient.

Necessary operations: one SQL select statement

4: $A \rightarrow L \rightarrow St$:$\left\{\{\{A_{id}\}, \{C_{id}\}\}_{\overline{K}_A}\right\}$

The patient decrypts each of the health care provider's identifiers C_{id} to identify them and subsequently selects which user, in our example, which health care provider she wants to revoke. Afterwards she returns the chosen health care provider identifiers and her identifier, which she encrypted with her inner symmetric key, to the storage via the logic. Finally the storage deletes the association from the user-relations table.

As all four ciphertexts, in other words, the patient's identifier encrypted with both inner symmetric keys and the health care provider's identifier as well encrypted with both inner symmetric keys, have been stored within one row, it is possible to revoke a related user without her presence.

Necessary operations: encrypt patient's identifier, one SQL delete statement for every users, who should be revoked

This workflow assures, that the patient is in full control of which health care provider is able to add medical data for her. Nevertheless it is still possible for the revoked health care providers to access datasets they have been granted. We show authorizing and revoking of single or groups of datasets in the Sections 5.3.7 and 5.3.8.

In the prior sections we introduced the workflows of how users can be added to the system and showed the set-up of relationships between them. With this basis we can consequently add pseudonymized data to the system.

5.3.3 Adding Medical Data to the System

All medical data in our approach is separated from the identification data to assure users' privacy. Therefore, all datasets are associated with j pseudonyms. Every pseudonym is unique for any patient-health care provider-medical data combination. Moreover, a so-called root pseudonym ψ_{i_0} exists, which is only related with the data owner, in our case the patient and the i^{th} anamnesis, diagnosis or treatment dataset.

1a: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

1b: $C \rightarrow L$:$\{f_{authenticate}(C_{id}) =?\}$

The patient and the health care provider authenticate against the system and establish each a secured channel between the logic and the user.

Necessary operations: mutual authentication for both users

2a: $A \rightarrow L \rightarrow St$:$\left\{\{A_{id}\}, \left\{\{A_{id}\}_{\overline{K}_A}\right\}\right\}$

2b: $C \rightarrow L \rightarrow St$:$\left\{\{C_{id}\}, \left\{\{C_{id}\}_{\overline{K}_C}\right\}\right\}$

Afterwards, the patient initiates the processing by sending her identifier to the logic. The health care provider also sends her identifier to the logic. The logic forwards both identifiers to the storage. Moreover, both participants transmit their identifiers encrypted with their particular inner symmetric keys. Thus, the logic is able to lookup the storage, if a relation has already been setup for these two actors (cf. Section 5.3.1).

Necessary operations: encrypt two identifiers, one SQL select statement

3a: $St \rightarrow L$:$\left\{\widehat{K}_C\right\}$

3b: $St \rightarrow L$:$\left\{\widehat{K}_A\right\}$

The storage replies with the particular inner public keys of the communication partners.

Necessary operations: two SQL select statements

4a: $L \rightarrow C$:$\left\{\left\{\{A_{id}\}, \{\psi_{i_j}\}\right\}_{\widehat{K}_C}\right\}$

4b: $L \rightarrow A$:$\left\{\left\{\{C_{id}\}, \{\psi_{i_0}\}, \{\psi_{i_j}\}\right\}_{\widehat{K}_A}\right\}$

The logic encrypts the patient's identifier with the health care provider's inner public key and sends it to the health care provider. Furthermore the logic generates two new random numbers, which represent the root pseudonym ψ_{i_0} and ψ_{i_1}, the pseudonym which will be shared between the health care provider and the patient. The logic encrypts these pseudonyms with the particular inner private keys and sends the needed ciphertexts to the health care provider and the patient.

Necessary operations: encrypt identifier, generate two new unique random numbers, one encrypt operation for the patient related pseudonym and two encrypt operations for the pseudonym, which will be shared between the patient and the health care provider

5: $C \rightarrow L$:$\left\{ \left\{ \{A_{id}\}, \{C_{id}\}, \left\{\psi_{i_j}\right\}, \{\tau_v\} \right\}_{\overline{K}_C} \right\}$

6: $C \rightarrow L$:$\left\{ \left\{ f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1}) \right\}_{\overline{K}_C}, \{\varphi_i\}, f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1}) \right\}$

The health care provider firstly decrypts the patient's identifier and the pseudonym ψ_{i_j} with her inner private key. Secondly, she begins to form the message for adding the new medical data by appending the anamnesis, diagnosis or treatment data in plaintext. Afterwards, she encrypts the pseudonym, the patient's and the health care provider's identifier as well as the related chosen tags τ_v with the health care provider's inner symmetric key.

For integrity purposes the health care provider, as the data enterer, has to ensure the medical data's integrity. Therefore, she signs a hash of the anamnesis, diagnosis or treatment data with her inner private key and encrypts this signature with her inner symmetric key. Finally, she transmits this message, containing the encrypted signed hash value, the signed hash value in plaintext and the medical data φ_i to the logic.

Further integrity issues may arise, as an attacker may inject unauthorized encrypted identifiers or tags into the database. Nevertheless, this is no security issue, because only authorized users are able decrypt the pseudonyms which are necessary to retrieve the matching medical data. Once more, logging and monitoring is the most appropriate way to notice that kind of prohibited behavior.

Necessary operations: decrypt identifier and pseudonym; encrypt tags, patient's as well as health care provider's identifiers and pseudonym; sign calculated hash, encrypt signed hash value

7: $L \rightarrow A$:$\left\{ \left\{ f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1}) \right\}_{\widehat{K}_A} \right\}$

The logic encrypts the plaintext version of the health care provider's signature with the patient's inner private key and forwards this ciphertext to the patient.

Necessary operations: encrypt signature

8: $A \rightarrow L$:$\left\{\left\{\{A_{id}\},\{C_{id}\},\{\psi_{i_0}\},\{\psi_{i_1}\},\{\tau_v\},\left\{f_{sign}(f_{hash}(\varphi_i),\widehat{K}_C^{-1})\right\}\right\}_{\overline{K}_A}\right\}$

9: $A \rightarrow L$:$\left\{\left\{f_{sign}(f_{hash}(\psi_{i_0},\psi_{i_j}),\widehat{K}_A^{-1})\right\}_{\overline{K}_A}\right\}$

The patient decrypts her opposite's identifier, the health care provider's signature and both pseudonyms ψ_{i_j}, ψ_{i_0} with her inner private key. Afterwards she sends her chosen tags, the signed data hash, the patient's as well as the health care provider's identifier and the pseudonyms, all encrypted with the patient's inner symmetric key, to the logic.

Furthermore, the patient, as the data owner, calculates a hash value over all pseudonyms and signs this hash. This measurement assures the integrity of the plaintext pseudonyms in the medical data table. She transfers this signature encrypted with her inner symmetric key for storage to the logic. Please note, the patient is the only user who holds an encrypted version of this integrity attribute because other users are not able to commit authorizing operations.

Necessary operations: decrypt identifier, signature and pseudonyms; encrypt tags, signature, patient as well as health care provider identifiers and pseudonyms; calculate and sign hash over pseudonyms

10: $L \rightarrow St$:$\left\{\left\{\{\psi_{i_j}\},\{\tau_v\},\{A_{id}\},\{C_{id}\},\left\{f_{sign}(f_{hash}(\varphi_i),\widehat{K}_C^{-1})\right\}\right\}_{\overline{K}_C}\right\}$

11: $L \rightarrow St$:$\left\{\left\{\{\psi_{i_0}\},\{\psi_{i_j}\},\{\tau_v\},\{A_{id}\},\{C_{id}\},\left\{f_{sign}(f_{hash}(\varphi_i),\widehat{K}_C^{-1})\right\}\right\}_{\overline{K}_A}\right\}$

12: $L \rightarrow St$:$\left\{\{\psi_{i_0}\},\{\psi_{i_j}\},\{\varphi_i\}\right\}$

13: $L \rightarrow St$:$\left\{\left\{f_{sign}(f_{hash}(\psi_{i_0},\psi_{i_j}),\widehat{K}_A^{-1})\right\}_{\overline{K}_A}\right\}$

The logic transfers the anamnesis, diagnosis or treatment data φ_i and a plaintext version of the pseudonyms to the medical data database in the storage. Afterwards the logic saves the encrypted tags, patient's and health care provider's identifiers as well as the encrypted pseudonyms in the pseudonyms table of the storage. Thus, this table can then be used to execute SQL select statements with the ciphertexts of, for example the patient's identifier and her chosen tags to gain data access.

Necessary operations: three SQL insert statements

Every participant of a medical dataset may hold different tags, in other words a keyword like x-ray or surgery, also called meta data or descriptive data (cf. [23,76] for a detailed description of a tags taxonomy). Consequently, if two health care providers are authorized to access a certain anamnesis, diagnosis or treatment dataset, they may apply tags that differ from each other as well as from the patient. As this descriptive information of the dataset needs to be hidden to reduce the possibility of profiling attacks and thus prevent guessing of a patient's identity, tags and other identifiers are stored encrypted with the particular users' inner symmetric keys.

To assure an appropriate runtime of the system, every SQL query is only conducted with ciphertexts to minimize the encrypting operations of meta data. Even if it is possible to select anamnesis, diagnosis or treatment data by the identifiers of the participants, this means that the tags have to be chosen carefully to achieve more appropriate results on the retrieval of datasets and therefore optimize the runtime.

Moreover, the medical datasets' timestamps need to be hidden. We propagate splitting up timestamps into their atoms. For example May 02, 2007 would fall basically into the tags May, 02 and 2007. This was a $Wednesday$, which is another tag. If we also add the week of the year, in our example the $week\_14$, the time ranges become query able as well.

A practical example would be a radiologist who invites patients for a check-up a week after the surgery. To receive a list of all patients who have been screened in week 13 and need to reattend a week later, she forms a query of her encrypted identifier and of the tags $knee$, $x-ray$, $needs\_follow\_up$, $Thursday$ and $week\_13$ The encrypted pseudonyms and their related meta data will be returned encrypted with the health care providers inner symmetric keys. Afterwards it is possible for her to select the desired medical datasets and retrieve the data as described in the next section.

Additionally every user may flag one dataset with an emergency pseudonym ψ_{i_ϵ}. This emergency information ϵ_A can be retrieved with a different authentication technique, which enables ad-hoc access for emergency doctors ED. We explain the method for emergency data access in Section 5.4.

5.3.4 Retrieving Medical Data from the System

There are different possibilities to retrieve a medical dataset. First of all the patient or a relative who has access to the patient's inner private key $\widehat{K}_A^{-1}$ are able to decrypt the patient's inner symmetric key. Hence, both are able to query the storage via the logic by encrypting the necessary tags, like keywords or a time-stamp in combination with an encrypted version of the identifiers, to look up a certain anamnesis, diagnosis or treatment by the usage of the root pseudonym.

Health care providers, who are only authorized to access non-root pseudonyms have to use their inner symmetric key to establish a connection with the medical dataset.

Furthermore, it is possible to hand over a couple of pseudonymized datasets to a research institution. In the following, we present the workflow when a patient wants to see a specific medical dataset.

1: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

The patient uses the authentication workflow to log on to the system.

Necessary operations: mutual authentication

2: $A \rightarrow L \rightarrow St$:$\left\{\{\{\tau_v\}, \{A_{id}\}, \{C_{id}\}\}_{\overline{K}_A}\right\}$

The patient prepares the where clause in the SQL statement by encrypting chosen tags, for example a keyword, time stamp or health care provider's identifier. The patient transfers the query to the storage via the logic.

Necessary operations: encrypt patient's identifier and desired tags

3: $St \rightarrow L \rightarrow A$:$\left\{\{\psi_{i_0}\}_{\overline{K}_A}\right\}$

If the query produced any results, the storage replies with a minimum of one or a set of encrypted root pseudonyms which the logic forwards to the patient.

Necessary operations: one SQL select statement

4: $A \rightarrow L \rightarrow St$:$\{\psi_{i_0}\}$

5: $St \rightarrow L \rightarrow A$:$\left\{\{\varphi_i\}, \left\{\left\{\left\{f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1})\right\}, \{\tau_v\}, \{C_{id}\}\right\}_{\overline{K}_A}\right\}\right\}$

The patient selects from the received list of pseudonyms, decrypts the desired pseudonym/s with her inner symmetric key and queries the logic with the plaintext pseudonym/s. The logic forwards the patient's request to the storage. The storage returns the matching anamnesis, diagnosis or treatment and their related encrypted signatures via the logic. Additionally the logic provides

the patient with all related tags of a certain pseudonym and the health care provider's identifier, even if they have not been within the query.

Necessary operations: one decrypt operation for the pseudonym and one SQL select statement for every desired medical dataset; decrypt signature, identifier and optionally the related tags for every medical dataset;

6: $A \rightarrow L \rightarrow St{:}\{C_{id}\}$

After decrypting, the patient queries the storage via the logic with the health care provider's identifier to receive the particular health care provider's inner public key.

Necessary operations: a maximum of one SQL select statement for every medical dataset

7: $St \rightarrow L \rightarrow A{:}\left\{\widehat{K}_C\right\}$

Besides the confidentiality, which is assured by pseudonymization, the patient proofs the dataset's integrity by re-calculating the hash-value and checking the validity of the health care provider's signature on the hash value. The signature can be verified, after decryption, by the usage of the health care provider's public key.

Necessary operations: calculate hash value and check validity of signature

Besides the assurance of the common security attributes another outlier may occur because we use ciphertexts to query the database. This special case exists with a probability of more than 50% if more than $2^{\frac{n}{2}}$ messages are calculated, whereas n is the length of the block length of the applied cipher. If we use for example AES with a block length of 128 bit for the inner symmetric keys, the probability that two different plaintext messages on which we apply two different keys, will result in the same ciphertext is more than 50% after 2^{64} messages. Nevertheless, this circumstance will not decrease the security of the system, but might lead to usability problems like error-messages in very rare cases.

As we already mentioned, our system is not only capable of primary, but also of secondary usage of the stored medical data. If no relation between more datasets is required, the researchers are authorized by adding another pseudonym, which the researcher and the patient share. This is comparable to the authorization of a health care provider for a medical dataset, but without the exchange of the patient's identifier, which will not be handed over to the researcher. If datasets

are handed over to a research institution it is necessary to verify which k-Anonymity (cf. Section 2.2.1) is given. This means of control is also important for follow-up studies to protect the participating patients' privacy.

In case it is necessary to base the results of a study on the medical history of the patients, it is also possible to use the same tag for a series of medical datasets. If the researchers want to invite the patients to a follow-up study or if new information about the studies' findings is available, they can add a flag to the specific anamnesis, diagnosis or treatment. This flag will be shown, for example in form of a dialog window, the next time a health care provider, authorized for this medical dataset or the patient herself authenticates against the system.

5.3.5 Updating Medical Data in the System

Every participant may specify a set of different tags for every anamnesis, diagnosis or treatment to reflect her needs and thus enhance the usability of the system. If the perception of the information may be changed over time or health care providers might want to add additional descriptive information to a medical dataset, then these datasets may have to be updated.

We present the workflow for the update process by a health care provider as an example.

1a: $C \rightarrow L$:$\{f_{authenticate}(C_{id}) =?\}$

1b: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

The health care provider and the patient authenticate against the system and establish a secured channel.

Necessary operations: mutual authentication

2: $C \rightarrow L \rightarrow St$:$\left\{\{\{\tau_v\}, \{C_{id}\}, \{A_{id}\}\}_{\overline{K}_C}\right\}$

Next, the health care provider chooses different tags or identifiers and encrypts them to query the storage via the logic. The search results can also be limited by adding a time-slot.

Necessary operations: encrypt identifier(s) and desired tags

3: $St \rightarrow L \rightarrow C$:$\left\{\{\psi_{i_j}\}_{\overline{K}_C}\right\}$

If the query produced any results, the storage replies with the encrypted pseudonyms which the logic forwards to the health care provider.

Necessary operations: one SQL select statement

4: $C \rightarrow L \rightarrow St{:}\left\{\psi_{i_j}\right\}$

5: $St \rightarrow L \rightarrow C{:}\left\{\left\{\varphi_i\right\}, \left\{\left\{f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1})\right\}, \left\{\tau_v\right\}, \left\{A_{id}\right\}\right\}_{\overline{K}_C}\right\}$

The health care provider selects medical datasets from the received list of pseudonyms by decrypting the desired pseudonym/s with her inner symmetric key and queries the storage via the logic with the plaintext pseudonym/s. The storage returns the matching anamnesis, diagnosis or treatment data and their encrypted signed hashes via the logic. Furthermore, the health care provider retrieves the data owner's encrypted identifier and all associated tags, even if they have not been in the query.

Necessary operations: decrypt ψ_{i_j}, one SQL select statement for every desired medical dataset

6: $C \rightarrow L{:}\left\{\left\{\left\{A_{id}\right\}, \left\{C_{id}\right\}, \left\{\psi_{i_j}\right\}, \left\{\tau_v'\right\}\right\}_{\overline{K}_C}, \left\{A_{id}\right\}\right\}$

7: $C \rightarrow L{:}\left\{\left\{f_{sign}(f_{hash}(\varphi_i'), \widehat{K}_C^{-1})\right\}_{\overline{K}_C}, \left\{\psi_{i_j}\right\}, \left\{\varphi_i'\right\}, f_{sign}(f_{hash}(\varphi_i'), \widehat{K}_C^{-1})\right\}$

The health care provider optionally updates the encrypted tags and the plaintext content and sends the changed entries to the storage. To assure the integrity of the changed data the health care provider confirms the retrieved signed hash. Then she calculates the new hash values and signs it. Moreover, she encrypts this value with her inner symmetric key and sends both the ciphertext as well as the plaintext signature to the logic.

Necessary operations: verify own signed hash value, calculate new hash value, sign and encrypt it

8: $L \rightarrow St{:}\left\{\left\{A_{id}\right\}, \left\{\left\{A_{id}\right\}, \left\{C_{id}\right\}_{\overline{K}_C}\right\}\right\}$

Prior update the logic checks, if the health care provider and the patient are still related by conducting a select query against the relations table. Then the logic queries the storage for the patient's inner public key.

Necessary operations: one SQL select statement to verify the relation, one SQL select statement

to retrieve the patient's inner public key

9: $St \rightarrow L$:$\left\{\widehat{K}_A\right\}$

10: $L \rightarrow A$:$\left\{\left\{f_{sign}(f_{hash}(\varphi_i'), \widehat{K}_C^{-1})\right\}_{\widehat{K}_A}\right\}$

The storage replies with the patient's inner public key and informs the logic if the patient and the health care provider are still related. In the following, the logic encrypts the plaintext version of the health care provider's signed hash and transfers this ciphertext to the patient.

Necessary operations: encrypt signed hash

11: $A \rightarrow L \rightarrow St$:$\left\{\left\{f_{sign}(f_{hash}(\varphi_i'), \widehat{K}_C^{-1})\right\}_{\overline{K}_A}\right\}$

The patient decrypts the received ciphertext and encrypts it again with her inner symmetric key. She subsequently sends this integrity-related attribute to the storage via the logic.

Necessary operations: decrypt signed hash with private key, encrypt again with symmetric key

12: $L \rightarrow St$:$\left\{\left\{\psi_{i_j}\right\}, \left\{\varphi_i'\right\}, \left\{\left\{f_{sign}(f_{hash}(\varphi_i'), \widehat{K}_C^{-1})\right\}, \left\{\tau_v'\right\}\right\}_{\overline{K}_C}\right\}$

Finally, the logic updates the changes of the medical data itself and the related encrypted signatures in the database.

Necessary operations: a maximum of two SQL update statement for every received medical dataset, one optionally for the tags and one for the medical dataset itself

With this workflow we assure the flexibility of our system which is necessary to assure appropriate queries for the life-long storage of patients' medical data. We also guarantee the integrity of updated anamnesis, diagnosis and treatment data by the usage of signed hash-values of the current medical data.

5.3.6 Deleting Medical Data from the System

It is not common to delete anamnesis, diagnosis or treatment datasets from an EHR system, but this can still be done on the patient's request. Our role-based access control models assures that only data owners who possess the root pseudonym are able to conduct delete operations. We therefore present how a patient is able to delete a certain medical dataset.

1: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

The patient uses the authentication workflow to log on to the system.

Necessary operations: mutual authentication

2: $A \rightarrow L \rightarrow St$:$\left\{\{\{\tau_v\}, \{C_{id}\}, \{A_{id}\}\}_{\overline{K}_A}\right\}$

Firstly, the patient selects different tags or identifiers and sends an encrypted version to the logic which forwards the whole statement to the storage.

Necessary operations: encrypt identifier(s) and desired tags

3: $St \rightarrow L \rightarrow A$:$\left\{\{\psi_{i_0}\}_{\overline{K}_A}\right\}$

If any matching results have been found, the storage returns the results via the logic to the patient.

Necessary operations: one SQL select statement

4: $A \rightarrow L \rightarrow St$:$\{\psi_{i_0}\}$

5: $St \rightarrow L \rightarrow A$:$\{\varphi_i\}$

The patient decrypts the received pseudonyms list and select the datasets she wants to view. Afterwards, the query will be forwarded by the logic to the storage which provides the related medical dataset.

Necessary operations: decrypt ψ_{i_0}, one SQL select statement for every desired medical dataset

6: $A \rightarrow L \rightarrow St$:$\left\{\{\psi_{i_0}\}, \{\psi_{i_0}\}_{\overline{K}_A}\right\}$

The storage deletes all pseudonyms and the related medical dataset entries upon receipt.

Necessary operations: one SQL delete statement for every received root-pseudonym

In the last sections we presented the basis data processing workflows of our system. The ability of an EHR system is based on data sharing [40, 41, 100]. Thus we now present the necessary authorizing and revoking operations.

5.3.7 Authorizing for Data Access

With the workflow add medical data to system (cf. Section 5.3.3) the data owner, in our case the patient and the associated health care provider, will be automatically assigned to a new anamnesis, diagnosis or treatment. If there is the need to authorize an additional user, another pseudonym has to be linked with the certain medical dataset, which the data owner and the additional user hold together.

We present an example in which the patient authorizes an additional health care provider.

1a: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) =?\}$

1b: $C' \rightarrow L$:$\{f_{authenticate}(C'_{id}) =?\}$

The patient and the health care provider, who shall be authorized, authenticate against the system and establish each a secured channel to the logic.

Necessary operations: mutual authentication for both users

2: $A \rightarrow L \rightarrow St$:$\left\{\{\{\tau_v\}, \{A_{id}\}, \{C_{id}\}, \}_{\overline{K}_A}\right\}$

Firstly, the patient encrypts her identifier A_{id}, chosen tags τ_v and optionally an already authorized health care provider's identifier C_{id} with her inner symmetric key $\overline{K}_A$ to query the storage via the logic.

Necessary operations: encrypt identifier(s) and desired tags

3: $St \rightarrow L \rightarrow A$:$\left\{\{\psi_{i_0}\}_{\overline{K}_A}\right\}$

If the query produced any results, the storage replies with a minimum of one or a set of encrypted root pseudonyms which the logic forwards to the patient.

Necessary operations: one SQL select statement

4: $A \rightarrow L \rightarrow St$:$\{\psi_{i_0}\}$

5: $St \rightarrow L \rightarrow A$:$\left\{\{\varphi_i\}, \left\{f_{sign}(f_{hash}(\varphi_i, \widehat{K}_C^{-1})\right\}_{\overline{K}_A}\right\}$

6: $St \rightarrow L \rightarrow A$:$\left\{\left\{\{C_{id}\}, \left\{f_{sign}(f_{hash}((\psi_{i_0}, \psi_{i_j}), \widehat{K}_A^{-1})\right\}\right\}_{\overline{K}_A}\right\}$

7: $C' \rightarrow L \rightarrow A$:$\{C'_{id}\}$

Afterwards, the patient decrypts the associated root pseudonym and selects the datasets she wants to share with another health care provider. Before sending the related root pseudonym to the logic, the patient verifies her signed hash value over the pseudonyms. As discussed in Section 5.3.3, this hash is based on all authorized pseudonyms in plaintext and has been signed by the patient to assure the authorization's integrity. Thus, to identify unauthorized changes in the database, the patient firstly decrypts her signed hash with her inner symmetric key and afterwards re-calculates and re-signs it. Moreover, the new health care provider transfers her identifier via the logic to the patient.

Necessary operations: decrypt ψ_{i_0} and signed pseudonyms hash, re-calculate and re-sign hash, compare hash-values, one SQL select statement for every desired medical dataset

8: $A \rightarrow L$:$\left\{\{A_{id}\}, \{C_{id}\}, \{C'_{id}\}, \{\{A_{id}\}, \{C'_{id}\}\}_{\overline{K}_A}\right\}$

In addition, the patient sends the identifier of all participating actors to the logic. These are (i) the patient's identifier, (ii) the identifier of the health care provider who she wants to authorize, and also (iii) the identifier of the health care provider, who has been the data-enterer. The logic needs all three attributes in plaintext for further processing. Moreover, the patient encrypts the new health care provider's identifier, which she previously received from the logic and her identifier, with her inner symmetric key, and then transfers these ciphertexts to the logic.

Necessary operations: decrypt data enterer's identifier and encrypt all three participating iden-

tifiers for every combination of the participating users

9: $L \rightarrow St$:$\left\{\left\{\{A_{id}\}, \{C'_{id}\}\right\}_{\overline{K}_A}, \{A_{id}\}, \{C_{id}\}, \{C'_{id}\}\right\}$

10: $St \rightarrow L$:$\left\{\left\{\widehat{K}_A\right\}, \left\{\widehat{K}_C\right\}, \left\{\widehat{K}'_C\right\}\right\}$

The logic checks the relation between the patient and the health care provider C' with a lookup on the encrypted relations table in the storage. Furthermore, the logic queries the storage for the users' public keys with the plaintext identifiers.

Necessary operations: one SQL select statement to verify an existing relation between the patient and the new health care provider, one SQL select statement for every participating user

11: $L \rightarrow A$:$\left\{\widehat{K}_C\right\}$

12a: $L \rightarrow A$:$\left\{\left\{\psi_{i_{j+1}}, C'_{id}\right\}_{\widehat{K}_A}\right\}$

12b: $L \rightarrow C'$:$\left\{\left\{\psi_{i_{j+1}}, A_{id}\right\}_{\widehat{K}'_C}\right\}$

The logic forwards the data enterer's inner public key to the patient.

Furthermore, the logic forms a new pseudonym $\psi_{i_{j+1}}$, which will only be shared between the patient A and the new health care provider C', and encrypts it with the inner public keys of both participants. Afterwards the logic transfers these ciphertexts, as well as an encrypted version of the particular identifiers, to the patient and the health care provider.

Necessary operations: encrypt new pseudonym ψ_{i_j} and the particular identifiers twice for every medical dataset

13: $A \rightarrow L$:$\left\{f_{sign}(f_{hash}(\varphi_i, \widehat{K}_C^{-1})\right\}$

The patient uses the data enterer's inner public key to assure the data's integrity by verifying the signed hashed data value. In case of an success, she forwards the gained plaintext signature to the logic.

Necessary operations: decrypt signed hash value, verify signature for every selected medical dataset

14: $L \rightarrow C'$:$\left\{\left\{f_{sign}(f_{hash}(\varphi_i, \widehat{K}_C^{-1})\right\}_{\widehat{K}'_C}\right\}$

15: $C' \rightarrow L$:$\left\{\left\{f_{sign}(f_{hash}(\varphi_i, \widehat{K}_C^{-1})\right\}_{\overline{K}'_C}\right\}$

The logic applies the new health care provider's inner public key and sends the ciphertext to her. Upon receipt, C' uses her inner private key and decrypts the message. Subsequently, she encrypts the signed hash with her inner symmetric key and returns it to the logic.

Necessary operations: encrypt signed hash value twice, decrypt once for every medical dataset

16: $A \rightarrow L$:$\left\{\left\{\{A_{id}\}, \{C'_{id}\}, \{\psi_{i_0}\}, \left\{\psi_{i_{j+1}}\right\}\right\}_{\overline{K}_A}, \{\psi_{i_0}\}, \left\{\psi_{i_{j+1}}\right\}\right\}$

17: $A \rightarrow L$:$\left\{\left\{f_{sign}(f_{hash}(\psi_{i_0}, \psi_{i_j}, \psi_{i_{j+1}}), \widehat{K}_A^{-1})\right\}_{\overline{K}_A}\right\}$

18: $C' \rightarrow L$:$\left\{\left\{\{A_{id}\}, \{C'_{id}\}, \left\{\psi_{i_{j+1}}\right\}, \{\tau_v\}\right\}_{\overline{K}'_C}\right\}$

The patient as well as the health care provider decrypt the respective counterpart's identifier and then encrypt it and their own identifier with their inner symmetric key. Moreover, both participants decrypt the received pseudonym and encrypt it with their inner symmetric key. Afterwards they send all ciphertexts to the logic.

To assure the pseudonyms integrity, the patient calculates the new hash value, based on all plaintext pseudonyms and transfers the signed identifier encrypted with her inner symmetric key to the logic. In addition the patient also returns the combination of root pseudonym and new pseudonym in plaintext to the logic. The latter is necessary to update the pseudonymized data.

Necessary operations: decrypt the new pseudonym, the participating users' identifiers and subsequently encrypt the chosen tags, identifiers and pseudonym for both participants; only the patient calculates and signs the new hash-value

19: $L \rightarrow St$:$\left\{\left\{\psi_{i_{j+1}}, \tau_v, A_{id}, C'_{id}\right\}_{\overline{K}'_C}\right\}$

20: $L \rightarrow St$:$\left\{\left\{\psi_{i_{j+1}}, A_{id}, C'_{id}\right\}_{\overline{K}_A}\right\}$

21: $L \rightarrow St$:$\left\{\{\psi_{i_0}\}, \left\{\psi_{i_{j+1}}\right\}\right\}$

22: $L \rightarrow St$:$\left\{ \left\{ f_{sign}(f_{hash}(\psi_{i_0}, \psi_{i_j}, \psi_{i_{j+1}}), \widehat{K}_A^{-1}) \right\}_{\overline{K}_A}, \left\{ f_{sign}(f_{hash}(\varphi_i), \widehat{K}_C^{-1}) \right\}_{\overline{K}'_C} \right\}$

The logic appends the new pseudonym $\psi_{i_{j+1}}$ together with the encrypted tags and identifiers as well as a plaintext version of the pseudonym with the particular anamnesis, diagnosis or treatment in the storage. Moreover, the logic updates the signed hash in the database to assure integrity of the plaintext data.

Necessary operations: two SQL insert statements for every pseudonym and newly authorized actor, one SQL update statement for the signed hash value; all operations for every newly added pseudonym

This workflow is the basis for sharing data between various health care providers. In our role-based access control model we assure that only users who possess the root pseudonym are permitted to authorize other users for a certain medical dataset (cf. Section 4.3).

From a security point of view the confidentiality is assured as no user may access, for example, the tags of another user. Moreover, only the data owner knows the identity of all participating users, while the information of how many users are authorized to see the dataset is not disclosed to the health care providers. Integrity is also guaranteed, because the medical data and the related pseudonyms, which are the only attributes in plaintext, are secured by the hash values of the health care provider as the data enterer and the patient as the data owner.

5.3.8 Revoking Data Access

As we stated in the previous section, it is possible to authorize other users for datasets by appending new pseudonyms. Consequently, users can also be revoked by deleting one of these pseudonyms. To assure that the patient is in full control of her data, a root pseudonym exists, which is not shared with other users at all. In other words, this root pseudonym only belongs to the patient who is usually the data owner, and cannot be deleted. All other pseudonyms may be deleted even without the notice of the other user who holds the pseudonym. This means that a patient may revoke any authorized user but the patient cannot be revoked from her datasets.

We introduce the workflow to revoke a previous authorized user from of certain anamnesis, diagnosis or treatment by the patient.

1: $A \rightarrow L$:$\{f_{authenticate}(A_{id}) = ?\}$

The patient uses the authentication workflow to log on to the system.

Necessary operations: mutual authentication

2: $A \rightarrow L \rightarrow St$:$\{\{\tau\}, \{A_{id}\}, \{C_{id}\}\}$

Firstly, the patient encrypts her identifier A_{id}, chosen tags τ_v like keywords or a time stamp and optionally a health care provider's identifier C_{id} to query the storage via the logic.

Necessary operations: encrypt identifier(s) and desired tags

3: $St \rightarrow L \rightarrow A$:$\left\{\{\psi_{i_0}\}_{\overline{K}_A}\right\}$

If the query produced any results, the storage replies with this set of encrypted root pseudonyms via the storage.

Necessary operations: one SQL select statement for every medical dataset

4: $A \rightarrow L \rightarrow St$:$\{\psi_{i_0}\}$

The patient decrypts the pseudonyms with her inner symmetric key and selects the anamnesis, diagnosis or treatment for which she wants to revoke a certain pseudonym $\psi_{i_{j+1}}$. She forwards this list via the logic to the storage.

Necessary operations: one SQL select statement for every selected root pseudonym

5: $St \rightarrow L \rightarrow A$:$\left\{\{\varphi_i\}, \left\{\{\psi_{i_j}\}, \{\psi_{i_{j+1}}\}\right\}_{\overline{K}_A}\right\}$

6: $St \rightarrow L \rightarrow A$:$\left\{\left\{f_{sign}(f_{hash}(\psi_{i_0}, \psi_{i_j}, \psi_{i_{j+1}}), \widehat{K}_A^{-1})\right\}_{\overline{K}_A}\right\}$

The storage responds with the list of datasets, associated users and their related signed hash values. The latter are based on the list of pseudonyms and assure integrity of data, which is held in plaintext. The patient firstly checks the validity of her signature by re-calculating the hash and signing it again.

Necessary operations: decrypt medical data-related attributes, re-calculate and resign hash value for every selected dataset

7: $A \rightarrow L \rightarrow St$:$\left\{ \left\{ \{\psi_{i_0}\}, \left\{ f_{sign}(f_{hash}(\psi_{i_0}, \psi_{i_j}), \widehat{K}_A^{-1}) \right\}_{\overline{K}_A} \right\} \right\}$

8: $A \rightarrow L \rightarrow St$:$\left\{ \{\psi_{i_0}\}, \{\psi_{i_j+1}\} \right\}$

If successful, she selects which users she wants to revoke from a certain anamnesis, diagnosis or treatment. Afterwards she sends the tuples of root-pseudonyms ψ_{i_0}, related pseudonyms $\psi_{i_{j+1}}$ and updated signed pseudonyms hash to the storage via the logic. Both pseudonyms are transmitted in plaintext and ciphertext in order to remove them from both databases.

Upon receipt, the logic deletes the matching pseudonyms. Hence, after these actions the revoked users cannot access the datasets because the medical data is not associated with the pseudonym any more.

Necessary operations: one SQL delete and one SQL update statement for every medical dataset

In addition to the cryptographic security aspect that we made the root pseudonym only available to one user, we additionally added a rule to our role-based access control model to assure that it is only possible for users, which possess the root pseudonym, to revoke other users (cf. Section 4.3).

5.4 Emergency Data Access

Emergency data, like the blood group or allergies for some medication, can be life saving if recognized early enough. The first helpers, ambulance men or the emergency room staff have to know all information about an unconscious patient. Thus it is vital to grant secure and fast access to the emergency data which forms a special subset of medical data.

One requirement is that actors, such as those just mentioned, should have immediate access to data, but only after they have been authenticated against a strong security system. The strong authentication is necessary because emergency data may contain sensitive information, for instance that a patient is infected with the HI-virus, etc.

5.4.1 Emergency Pseudonym Generation Scheme

The transaction pseudonym generation scheme discussed in this section is used as basis for the authentication mechanism. We use HMAC to calculate the underlying pseudonyms (cf. Section

3.3.2).

In general, every emergency access pseudonym η_{U_i}, which is unique for each participant, is the result of an earlier cryptographic calculation that is based on its respective successor pseudonym $\eta_{U_{i-1}}$. The only exception is the calculation of the initial pseudonym η_{U_1}. This uses a random number or a random string r as input value which may be exchanged, even in clear text, between the communication parties, because only the key needs to be concealed.

In our system, the logic generates and handles keys and pseudonyms and therefore forms a trusted instance. Regarding the workflow of viewing data we define the role of an "emergency doctor", that has access to this information. The constraints are that any user of this group is able to view but not manipulate, the emergency data by entering a card-number. An outdoor wearable augmented reality collaboration system (OWARCS cf. [142]) is used to communicate with the logic and to form the necessary pseudonyms. This applies for the user's device as well as for any OWARCS equipped with the necessary keys and a crypt-module, which is able to conduct HMAC pseudonym calculations.

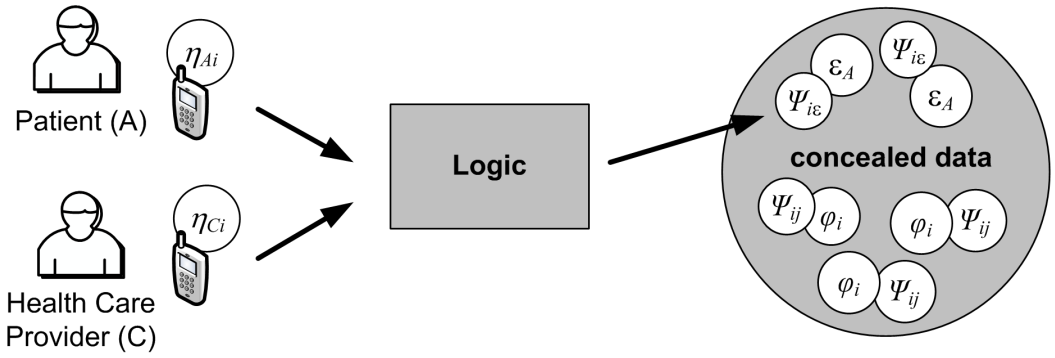

Figure 17: Emergency Workflow

5.4.2 Retrieve Emergency Data

As depicted in Figure 17 the device generates a unique emergency access pseudonym η_{C_i} and sends it to the logic. The logic verifies this pseudonym and queries the concealed hull for the patient's related emergency data pseudonym ψ_{i_ϵ} to look up the patient's associated emergency data ϵ_A. We now introduce the workflow steps in detail.

1: $ED \rightarrow L: \left\{ \{\{\eta_{C_i}\}, \{ED_{id}\}, \{A_{id}\}\}_{\widehat{K}_L} \right\}$

First of all the emergency doctor generates a unique emergency access pseudonym η_{C_i} which unambiguously identifies the emergency doctor $ED \in \mathcal{ED} \subset \mathcal{C}$ and encrypts this pseudonym and

the patient's identifier, which is printed on a card with the logic's inner public key, which is then stored in the emergency device. Afterwards she sends both ciphertexts to the logic.

Necessary operations: generate emergency access pseudonym, encrypt pseudonym and patient's identifier

2: $L \rightarrow St$:$\{ED_{id}\}$

3: $St \rightarrow L$:$\left\{\{\eta_{C_i}\}, \left\{\widehat{K}_C\right\}\right\}$

Upon receipt, the logic decrypts the message and queries the storage with the emergency doctor's identifier for her current access pseudonym $\eta_{C_{i'}}$. If both pseudonyms match, the doctor has been identified. Moreover, the logic checks the user type because only emergency doctors have the right to retrieve emergency data for others. Patients may only receive their own emergency data. The storage replies with the health care provider's inner public key.

Necessary operations: two SQL select statements

4: $L \rightarrow St$:$\left\{\{ED_{id}\}, \{\eta_{C_{i+1}}\}\right\}$

After successfully authenticating, the logic forms the next emergency access pseudonym $\eta_{C_{i+1}}$ with the appliance of HMAC and updates it in the database.

Necessary operations: one HMAC calculation, one SQL update statement

6: $L \rightarrow St$:$\{A_{id}\}$

7: $St \rightarrow L$:$\{\psi_{i_\epsilon}\}$

Afterwards the logic queries the database for the patient's emergency data pseudonym ψ_{i_ϵ}. - We described in Section 5.3.3, how a patient may define a pseudonym as emergency data pseudonym.

Necessary operations: one SQL select statement

8: $L \rightarrow St$:$\{\psi_{i_\epsilon}\}$

9: $St \rightarrow L$:$\{\epsilon_A\}$

The related emergency data ϵ_A is retrieved from the storage and returned to the logic.

Necessary operations: one SQL select statement

$$10:\ L \rightarrow Ed: \left\{ \left\{ \{\epsilon_A\}, \left\{ f_{sign}(f_{hash}(\psi_{i_\epsilon}), \widehat{K}_L^{-1}) \right\}_{\widehat{K}_C} \right\} \right\}$$

The logic uses the emergency doctor's inner public key $\widehat{K}_C$ to encrypt the emergency data ϵ_A and forwards the ciphertext to the emergency doctor. Moreover, the logic calculates a signed hash value with her inner private key, based on the emergency data to assure data integrity. This proof also confirms the logic's identity for ED, because only the logic possesses its public key. This is a necessary step because the emergency doctor and the logic are communicating over an unsecure channel.

Afterwards, the emergency doctor decrypts the received emergency data with her inner private key $\widehat{K}_C^{-1}$ and verifies the signature by re-calculating the hash and proofing the signature with the logic's public key. Finally, the emergency doctor uses HMAC to form her next emergency access pseudonym $\eta_{C_{i+1}}$.

Necessary operations: decrypt emergency data, re-calculate hash, verify signature, one HMAC calculation

With this authorization mechanism replay attacks can be avoided as long as the user's HMAC password has not been stolen or compromised. Moreover, the emergency data consists only of small ascii-text-parts and can therefore be also decrypted on special mobile devices with small memory and computing power. This fact assures that the confidentiality of the data can be trusted, although an unsecure communication channel between the emergency doctor and the logic, for example the Internet, might be used. The integrity is assured by the usage of digital signatures. Appropriate availability can be guaranteed by the usage of two different channels like Wi-Fi (IEEE 802.11 technologies) and Universal Mobile Telecommunications System (UMTS).

As many patients may still regard their emergency data as sensitive medical information, non-repudiation can be achieved by a logging functionality combined with monitoring.

5.5 Comparison to Existing Approaches

In this section we provide a summary of the difference between PIPE and existing EHR approaches.

As we outlined in Section 3.8 a major security shortcoming of a pseudonymization system is to rely on a patient-pseudonyms list. This centralized list, which is used for example by Pommerening et al. [98, 99] or Thielscher et al. [141] (cf. Section 3.8.1 and 3.8.2) is used for re-identification purposes. Thus it opens a vulnerability for outsider and insider attacks. Another possibility proposed by Onuma et al. [94] is the integration of a re-identification module (cf. Section 3.8.4) in combination with a centralized key. Even if the access to this module has been secured by the usage of encryption, the key is accessible for a single administrator. Hence, regarding the security design principle of separation of privilege (cf. Section 3.4.6) this technique cannot be considered secure because one person with access to the key can associate the anamnesis, diagnosis or treatment data with the particular patients' identities.

In PIPE we circumvent this described security flaw by the usage of our layered architecture (cf. beginning of Section 5). Thus, we realize the access to the pseudonyms and subsequently the pseudonymized data by the appliance of a multi-level encryption (cf. Section 4.3). Moreover, we solve the authorization by sharing encrypted secrets. In detail, every user possesses a smart card with a so-called outer key pair. We assure the access only for permitted persons to the inner security tokens by encryption with the mentioned outer security token. After the inner keys have been decrypted, the inner keys subsequently allow access to the pseudonymized datasets.

Regarding the backup approach of lost or destroyed security tokens, Thielscher et al. [141] operate a dedicated computer system off-line. In case a patient needs a duplicate for her smart card, the card is re-produced by a administrator utilizing this computer. Though, this organizational workaround increases the security regarding attacks from outside the system, insider attacks are still possible. In Section 4.5 we discussed how a secure backup keystore can be established. We achieve a high level of security by the appliance of a threshold scheme [131]. As not only the security but also the costs are increased with the appliance of a threshold scheme, we introduced a variant of our secure backup keystore in Section 4.5.2.

Another attacking possible on an EHR system is the dependence on a single pseudonym as proposed by Pommerening et al. [98, 99]. In other words, every patient's dataset is associated with the same pseudonym. As an attacker is able to combine the single datasets to the medical history, profiling attacks may occur.

In PIPE we use distinct randomized pseudonyms for every patient-health care provider-medical dataset combination (cf. 5.3.3). In other words if two health care providers are authorized for a certain dataset, three pseudonyms exist for that specific dataset. The first pseudonym, the so-called root pseudonym is only held by the patient. Thus, this also allows us to easily revoke already authorized health care providers by simply deleting the specific health care provider's pseudonym.

In 2001 Schmidt et al. also proposed an approach which is based on multiple encryption operations to access the data [126] (cf. Section 3.8.5). In contrast to PIPE, they use only one encrypted identifier for every medical dataset. The keys to re-establish the patient's identity is based on the appliance of a single key which is used to get access to the file's encrypted identifier. This technique may lead to the result, that users who shall be revoked, may still have access if they wrote down the file's key. Thus compared to PIPE, in their approach expensive operations are necessary in case of revoking.

Regarding the emergency access to the patients' medical data, Peterson's approach (cf. Section 3.8.3) permits people with reading access to the patients' anamnesis, diagnosis or treatment data in case a global key is available. He describes that in case of an unconscious patient, the emergency doctor looks for a card that the patient should carry. On this card the global key is printed. He argues, that the usage of long keys would increase the security, because longer keys can hardly be remembered [96]. Nevertheless an attacker could make a copy or a photo of the patient's card. In contrast to the open system of Peterson, Thielscher et al. outlines the concept of a call center. Emergency doctors have to authenticate them via phone and will then receive the vital information [141].

In PIPE we use an ad-hoc authentication mechanism, which is based on a novel pseudonymization technique (cf. Section 5.4). The patient is able to define, which of her data will be marked as emergency data. In case of an emergency, the health care provider authenticates against an emergency mobile device. Afterwards, the device establishes a reliable and replay-resident connection with the logic. The logic returns the available emergency data.

Summary In this section we discussed the main contribution of this thesis, PIPE's novel formal workflow framework for the pseudonymization of medical data. We showed how the users and our backup keystore can be administrated. We introduced workflows for adding, retrieving and updating of data and stated the necessary steps for authorizing and revoking.

In the upcoming section we elaborate on our proof-of-concept prototype as feasibility study for our approach.

6 Proof of Concept Prototype

In this section we introduce the concepts and methods, which we used to implement the formal workflows in our prototype. Firstly, we discuss a practical example of the applied pseudonymization technique. Then in Section 6.2 we introduce the relational database model. Afterwards we sketch our prototype's architecture in Section 6.3. Then we discuss the outcomings of our feasibility study at Genosense Diagnostics[15)] in Section 6.4. — Genosense Diagnostics contributes to the long-term improvement of human welfare by individualized medicine due to the patients genetic profile. As these genome data is a unique finger print for every patient, it is necessary to assure secure storage and processing of that sensitive information. — Finally we introduce a process for the integration of PIPE in Section 6.5.

6.1 Pseudonymization

Tables 10, 11 and 12 present a simplified representation of the database structure of our approach. Please note, that the given views are not normalized to increase readability.

id	Name
1234	John Doe
5678	Dr. Friendly

Table 10: Identification Database

ψ_{i_j}	A_{id}	C_{id}	τ_v
$\{abc\}_{\overline{K}_A}$	$\{1234\}_{\overline{K}_A}$	$\{5678\}_{\overline{K}_A}$	$\{WeedsAllergenTest\}_{\overline{K}_A}$
$\{def\}_{\overline{K}_A}$	$\{1234\}_{\overline{K}_A}$	$\{5678\}_{\overline{K}_A}$	$\{WeedsAllergenTest\}_{\overline{K}_A}$
$\{def\}_{\overline{K}_C}$	$\{1234\}_{\overline{K}_C}$	$\{5678\}_{\overline{K}_C}$	$\{Parietariajudaica\}_{\overline{K}_C}$

Table 11: Anamnsis Database - Table Pseudonyms

ψ_{i_j}	$Test1$	$Test2$	$Test3$
$\{abc\}$	rPar j 1 = pos	rPar j 2 = neg	rPar j 3 = neg
$\{def\}$	rPar j 1 = pos	rPar j 2 = neg	rPar j 3 = neg

Table 12: Medical Database - Table Data

As aforementioned, two databases exist, one where the identification data is stored and another one which holds the anamnesis, diagnosis and treatment data as well as the related pseudonyms. Our example consists of one diagnosis which is associated with two pseudonyms. The root pseudonym (abc) is only known by the patient John Doe, whereas the second pseudonym (def)

[15)] `http://www.genosense.at/`

is held together with the health care provider Dr. Friendly. Both participants use different tags (WeedsAllergenTest, Parietariajudaica) for the diagnosis which are stored in the last table.

To conduct secure pseudonymization of medical data (cf. Section 3.9) is is necessary to conceal all relations between the actors and their data. In our example the patient and the health care provider use their inner symmetric keys in order to conceal the relation between them (1234, 5678) and John Doe's dataset by encryption. Furthermore the confidentiality of the meta data is also assured by encryption.

As discussed in Section 4.3 these inner keys are only accessible by the usage of the outer keys. Hence, only authorized persons, in other words users who hold an encrypted pseudonym are able to relate the pseudonymized medical dataset with her data owner.

Following this basics we provide a full database scheme in the next section.

6.2 Relational Database Scheme

As outlined in the previous section we use two different databases to separate the users' identification data from the pseudonymized datasets. Please note, that it does not matter if we would use only a single database, because the information security relies solely on the aforementioned techniques.

In this section we discuss the attributes of a relational database, which is the basis to realize PIPE as an additional security layer in a legacy application. This scheme is similar to the stand-alone client application, because instead of storing the data itself in the database, we use pointers, like primary or secondary keys from the legacy application, and store them pseudonymized. The original keys in the legacy application's database are removed. Hence, it is only possible to unveil the relation between a user and her datasets by the usage of the issued pseudonyms.

Our obligations regarding the introduction of an additional layer are not only based on the object orientated paradigms like encapsulation, this technique also eases the implementation and creates a light-weighted protocol. The latter further conforms to the security design principle of economy of mechanism (cf. Section 3.4.3). Moreover, it simplifies testing, which is often an important security aspect.

Therefore, regarding the identification database (cf. Figure 18), we do not store the users's identifying attributes but use the pointer to the legacy application's users table instead. This is expressed in the table *UserMapping*. The relation between the tables *Identification* and *UserMapping* also allows us to map for example the same person, who may use PIPE within two different

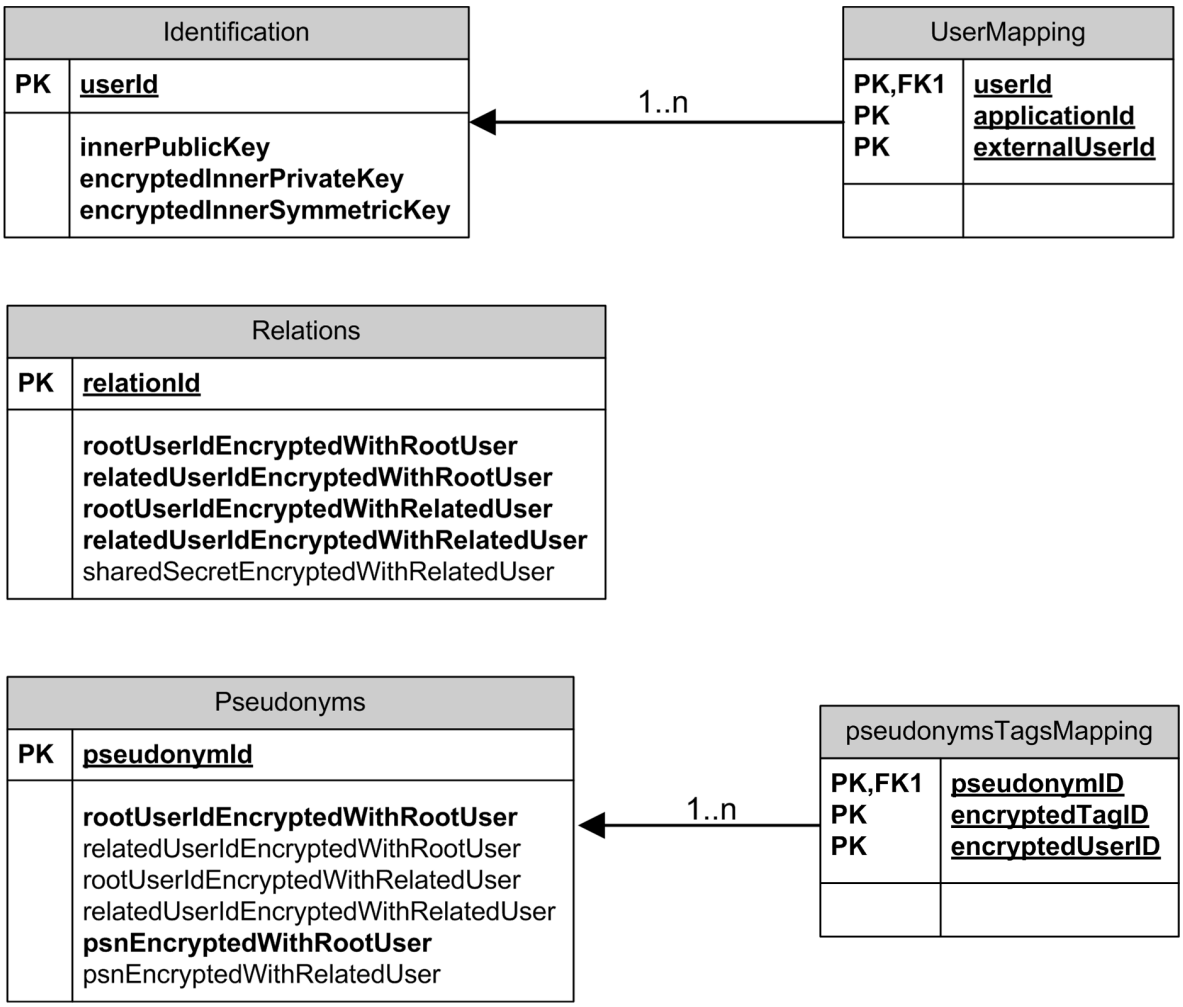

Figure 18: Database Diagram - Identification Data

applications to have only a single user account. Thus, she could log on to both applications with the same smart card and in addition share pseudonymized data between the applications, in case that kind of exchange is supported.

The table *Relations* represents the user-relations as discussed in the Sections 5.3.1 and 5.3.2. Both user ids are stored encrypted with the particular inner symmetric keys to make it possible for the participating users to query the database.

In the table *Pseudonyms* all pseudonyms are stored encrypted. Again, as we hold any pseudonym encrypted with the particular inner symmetric keys of the participants, all authorized persons may query the storage individually. In addition, the *pseudonymsTagsMapping* table represents the chosen tags or timestamps of each user. The $1 : n$ relation allows that every user may have n tags for any anamnesis, diagnosis or treatment dataset.

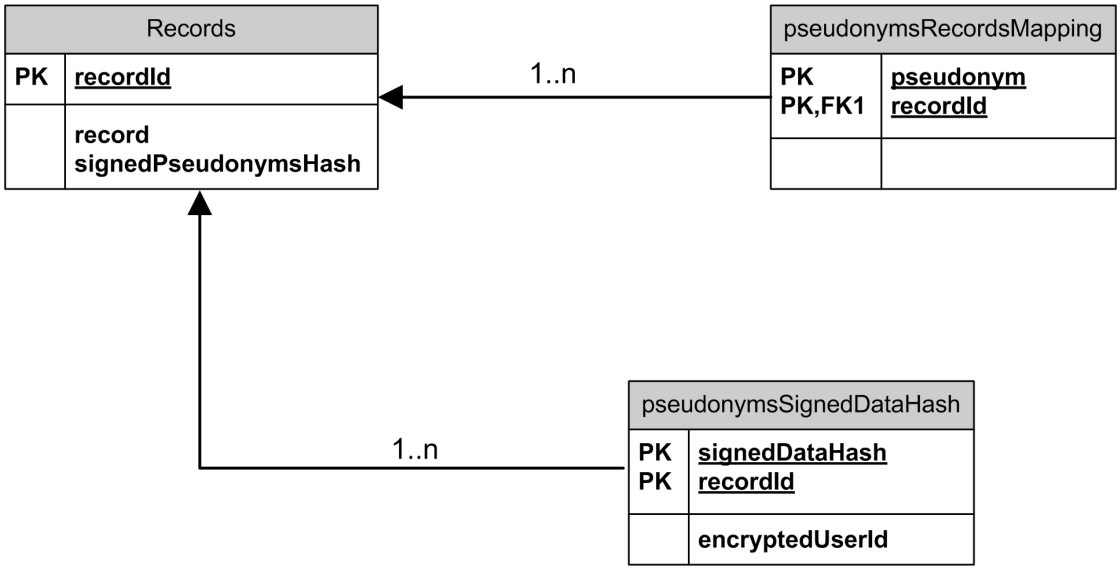

Figure 19: Database Diagram - Pseudonymized Medical Data

The medical database (cf. Figure 19), consists of three related tables. Firstly, the *Records* table holds the mapping attributes of the legacy application's records table. This would be, for example, the legacy application's primary keys of the medical data table.

In addition the hash-value over all plaintext pseudonyms which the patient signed and encrypted with her inner symmetric key is stored in *Records*. As previously mentioned (cf. Section 5.3.3), besides the root pseudonym, j other pseudonyms may be associated with one anamnesis, diagnosis or treatment, to equip health care providers with access rights. Thus, the table *pseudonymsRecordsMapping* holds all pseudonyms in plaintext. Please note, that it is not necessary, to encrypt the records or pseudonyms, because confidentiality is guaranteed by pseudonymization.

To assure the integrity of the medical data or the legacy application's mapping in the *Records* table, we store signed data hashes of each record. All participants of every medical dataset hold an encrypted version of these attributes. Hence, it is also possible for all associated users to prove a dataset's integrity by the usage of the *pseudonymsSignedDataHash* table.

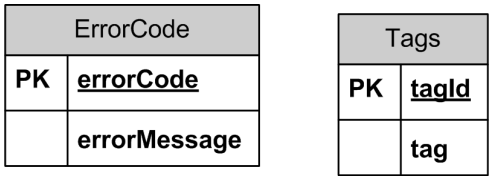

Figure 20: Database Diagram - General Tables

As depicted in Figure 20, we integrated two general tables besides the database tables which hold

the user's data. The *ErrorCode* table supports the developers with information, while the *Tags* table stores all available tags in plaintext. We introduced this measurement to ease selection of tags for the users. As someone might expect a security issue, this table can also be omitted without discontinuation of the PIPE service.

Our database system of choice is mySQL[16]. We decided to use it because of its scalability and the factor that mySQL is open-source.

Remark: A full database diagram of our proof-of-concept prototype will be given in the Appendix.

Based on this secure pseudonymized database structure, we show how we implemented the components of our formal workflow framework in the next section.

6.3 Architecture

Our architecture (cf. Figure 21) basically consists of a centralized service, the logic. The logic is securely connected to its database, which we called the storage in the formal sections. As aforementioned, it is necessary that the logic can trust the service and vice-versa. Hence, these two components are within a secured network area or connected over a secured channel, for example with the appliance of TLS. Moreover, we propagate the encryption on database level for additional security.

Another member of this trust circle is the key issuing component together with the personalization device for creating the smart cards. The access to this hardware security module (HSM) is secured with the appliance of a threshold scheme. Furthermore, the HSM also offers the functionality to create random values.

There are two possibilities of client software. Firstly, the client can be a stand-alone application. This type of client directly implements the necessary function-calls. Secondly, PIPE's functionality can also be added to an existing legacy software system. In that case, we use an API (Application Programming Interface) with an interface, which can be implemented in the legacy application.

The function calls are sent and received over SOAP (Service-oriented architecture protocol). SOAP, which was earlier referred to as 'Simple Object Access Protocol', is an XML-based protocol. In 2003, version 1.2 became a W3C (World Wide Web Consortium) recommendation [149]. There are several messaging patterns but Remote Procedure Calls (RPC) are most commonly

[16] `http://www.mysql.com/`

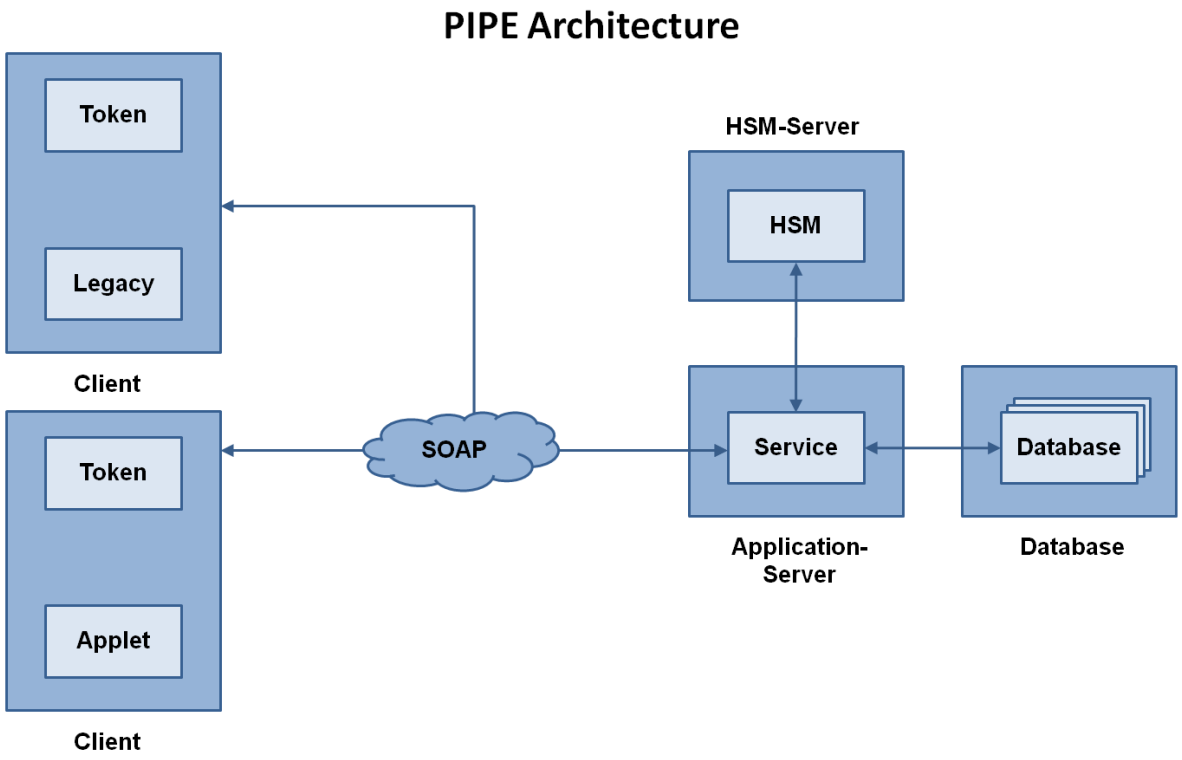

Figure 21: Overview of PIPE's Architecture

used. For an RPC, the client sends a message to service and receives an immediate reply. The calls are normally handled over the Internet Protocol Stack, with HTTP or HTTPS on top of it.

Every user possesses a security token which holds her outer key pair and identifier. In our prototype we equipped the users with smart cards. — This choice comes with another advantage. — All cryptographic operations on the client side can also be directly handled on the smart card. Thus, this reduces the risk of trojan horses or other virus attacks which may infect on a personal computer, because the smart card reader is, from a security point of view, encapsulated and separated from the computer. It is nearly impossible for viruses to steal, for example, the user's PIN code. Nowadays, smart cards readers can also be equipped with a display. Hence, the possibility of client-side man-in-the-middle attacks regarding the integrity of system messages is significantly lowered[17].

Due to the constraints of our business partner we used the Microsoft .NET framework[18] for the client-side API. This framework ships with a virtual machine on which the execution of programs

[17] http://www.a-trust.at/
[18] http://www.microsoft.com/NET/

is based. For the logic component we setup a JBoss Application Server[19] which is implemented in Java[20].

Summary In the last sections we discussed the architecture and the database scheme of our implementation. We showed how we realized our formal workflow framework for the secure pseudonymization of medical data.

In the following we provide further details on the realization and testing of PIPE in our feasibility study.

6.4 Feasibility Study

As aforementioned, PIPE can be implemented as stand-alone application or PAT for an existing system. All data queries and modifications are handled over our centralized service, the logic. Every pseudonymized application communicates with the logic over a local API. We implemented unit-tests for each function and formal workflow in PIPE's API to assure the continuous validity in case of extensions or corrections. The full function list of the API will be given in the Appendix.

To ease the integration of the additional security layer we minimized and generalized the function calls of PIPE's API. For security reasons we decided to avoid holding any data in the API's memory. Hence, the applications have to store and secure temporary outcomings of the workflow steps themselves. As support for the developers we issued template functions which are the basis to integrate PIPE's functionality into existing systems. This decision leads to an increase of the possible usage of PIPE, as the basic workflows can be re-ordered to also cover other areas of application.

Regarding the transport protocol we integrated a version number to ease step-wise integration into larger systems. Thus, upgrades can also be committed in smaller projects. Moreover, as the communication is based on XML, different confidentiality and integrity techniques as well as different server technologies are supported.

Before we integrated PIPE's functionality we planned the minimal requirements for the hardware. Besides a server we needed several client computers for our tests. After the first tests we measured about approximately nine seconds for one execution of an "adding data to the system" workflow. As this was clearly not sufficient, we tried to improve the performance of the resulting bottleneck, the security tokens.

[19] `http://www.jboss.org/`
[20] `http://www.java.com/`

Regarding the efficiency, we tested different smart cards and card readers. Our tests lead to varying results of 0.3 to nearly 1.1 seconds for processing a single encryption operation with a 2048bit RSA key. On some smart cards it is further possible to store the user's inner key pair as well as the user's inner symmetric key in the smart card's cache. Hence, the performance and efficiency of PIPE strongly varies with the underlying hardware.

In the following we outline the technical constraints of the hardware which led us to the best results.

- *Logic+Storage* CPU: 2x Intel Core 2 Duo 3.17GHz; RAM: 2048 MB ECC; Hard-drives: 3x 80 GB SAS as Raid 5 Array; HSM: nCipher nShield PCI 500[21)]
- *Clients* CPU: 2x Intel 2.00GHz; RAM: 1024 MB; Hard-drives: 1x 80 GB SATA
- *Security Tokens* Smart card reader: Reiner SCT cyberJack[22)] (the same smart card reader is commonly used for the Austrian Citizen Card[23)]); Smart card: Gemalto Classic TPC IS V2[24)], 36K EEPROM, Java v2

Our decision for the card-reader was two-folded. Besides the reason of the broad distribution in Austria, our card reader also has the ability to view small messages directly on its display. Hence, this increases the security, because it is harder for trojan viruses to interfere user-machine communication. The latter fact classifies this kind of smart card reader as class three which is also the capable for electronic banking and direct payment.

As security tokens we used Gemalto smart cards. This security token inherits an own crypt processor to conduct encrypting/decrypting and signing operations with 2048bit RSA. Moreover, it is capable to execute small Java applets. In future developments this could lead to another performance and security increase, because some workflow steps do not rely on user interaction and can therefore be executed automatically.

In that case, the user authorizes against her smart card by entering the PIN-code. Afterwards the keys are cached for a defined period of time. Hence, most of the information can be encrypted, decrypted, signed and verified automatically and directly in the smart card's memory. Nevertheless, this opens a vulnerability, because a cached session could be misused in the residual time period if the user does not properly logout of the system.

[21)] `http://www.ncipher.com/`
[22)] `http://www.reiner-sct.com/`
[23)] `http://www.buergerkarte.at/`
[24)] `http://www.gemalto.com/`

Together with our business partner we developed the proof-of-concept prototype with the implementation technique rapid prototyping. As first step we implemented a sketch of our system and applied the elaborated requirements and security constraints, circumventing security flaws of other systems. As aforementioned, we adapted the constraints so that users do not need to enter the PIN-code at every workflow step. After a successful authentication against the smart card we cached the user's keys for five minutes.

Then we measured the execution times of different workflows and decided in contrast to the original design, which was solely based on asymmetric encryption, to also include the usage of symmetric keys. This decision lead to a slower authentication process when the user logs on to the system, because she needs to decrypt one more key. Nevertheless, we also decreased the necessary execution time of nearly all workflows at the same time. After a re-work of our first prototype, we formalized, designed and released a second version which was based on these adapted workflows.

Appliance in a Real-World Environment Following excessive testing, we began to plan the deployment of PIPE in the network of Genosense Diagnostics. Their main business process prior to pseudonymization consists of the following steps:

1. A patient consults one of Genosense Diagnostics partners or laboratories to take a dna sample material buccal mucosa.

2. Afterwards a health care provider determines together with the patient which specific diseases of the fields (i) Gynecology, (ii) Obstetrics, (iii) Andrology resp. Urology, (iv) Cardiology or (v) Pharmacology should be researched.

3. The health care provider submits the sample together with the patient's identification form to Genosense Diagnostics.

4. An employee of Genosense Diagnostics creates or updates the patient's dataset.

5. Researchers of Genosense Diagnostics work out the desired results together with an overview on potential diseases and insert the results in their system.

6. Then an administrative employee of Genosense Diagnostics prints out the results and returns them to the health care provider, who informs the patient about the gained information.

7. The same department of Genosense Diagnostics also prints out and sends a bill to the

health care provider, who receives the payment directly from the patient.

8. In case other research needs to be conducted to increase the information about a certain disease, the health care provider can demand additional examinations on the patient's genome.

The goal of Genosense Diagnostics was to increase the security and efficiency of the existing information flows. In several workshops we firstly decided to shift their current combination of a paper-based system and a centralized electronical database at Genosense Diagnostics in a prototypical and additional three phases.

Firstly, the proof-of-concept prototype should be implemented to support a basic inner workflow, so that the users and developers of our business partner can test the stability of PIPE. After successful testing, we planed to increase the functionality of the Microsoft Access based system to include all paper-based workflows based on the functionality of PIPE as additional security module.

In the next step, Genosense Diagnostics CEOs plan to integrate their worldwide network of labors and partners. On a long-time progression the management thinks about the development and deployment of a web client to establish access to the medical information for their patients with their own smart card.

Following this road map, we worked out the first workflow which shall be secured by PIPE. The next enumeration shows the difference of the original and the pseudonymized basic workflow at Genosense Diagnostics after the prototypical phase.

1. In contrast to the original system, the employee at Genosense Diagnostics who creates or edits the dataset is only able to view or adopt the personal-related and aggregated medical data. As in that phase, the patients and the external health care providers do not receive cards, we integrated a role-based controlled HSM on behalf of these actors. Thus, it is also possible to set in the simulated patient as the data owner of every research dataset. In other words, as discussed in Section 5.3.3, she possesses the root pseudonym. In addition the health care provider who submitted the sample is automatically authorized for her patients.

2. We equipped every member of the health care team of Genosense Diagnostics with a smart card. If an employee conducts research for a patient, she is authorized for this specific datasets by adding another pseudonym.

3. After all examinations are finished, a medical employee prints out and returns the results to

the health care provider. The employee who issues the bill is again only permitted to access the personal-related and aggregated medical data. This is equivalent to the permissions in step one.

4. For the backup keystore (cf. Section 4.5), we implemented a system with three human operators whereas only two humans have to act together to re-establish a lost or destroyed smart card.

Following the workshops and the information we gained from our prototype, the users were satisfied with the optimized execution time of in average two seconds for the retrieval of pseudonymized dataset. Moreover, the employees of Genosense Diagnostics stated, that the gained security also increased the trust in their own system. We currently work on increasing the number of pseudonymized workflows to conclude the next step in the long-term plan.

6.5 Integration of PIPE

In the following we state the necessary steps to integrate PIPE into a legacy application with a minimum of possible downtime and the avoidance of possible pitfalls.

1. In the first step we determine the underlying workflows of the existing business system. Therefore we arrange meetings and workshops and work out a consent on the future workflows between all stakeholders.

2. As not all workflows might deal with personal-related data we decide which workflows should be pseudonymized. We further identify what users should be authorized for which information. One result of the workshops with the members of our business partners was that it is not always necessary that users have continuous access to the user's identity. For example the laboratories are able to conduct their research with anonymized data whereas the accounting department does not need access to the medical data. Hence, we specify in which workflow step what users or user groups need access to which subsets of the patient's data.

3. This requirements analysis is the input for workshops with the participating software engineers, the chief privacy officer and the decision makers. The goal of this step is to establish a consent of all stakeholders on the gained security, occurring costs and resulting usability. This step is important, because the existing trade-off between the constraints of the future system may also directly influence the user-acceptance and productivity of the system. For that reasons we implemented a spreadsheet which can be used to conduct "What-If" analyzes on the applied security and the occurring costs.

4. Afterwards we implement the novel PIPE framework in a copy of the existing software. In our case we set up a second server system and test clients to avoid interruptions in our business partner's daily work. Then we integrate the necessary function blocks and test the newly created PAT. We run tests together with our business partner to avoid misunderstanding and to develop unit-tests to ease future maintenance tasks. Moreover, we develop a migration scenario together with the technical staff of our business partner to avoid loss of data.

5. The next step is to set-up the user-accounts and to produce the smart cards for all authorized users and operators.

6. In parallel we hold courses to achieve immediate user acceptance for the added security layer.

7. After excessive usage-based testing (cf. [158]) we define a migration date and inform the users about the offline status of their system.

8. On the migration date we replace the old system with the newly created privacy-enhanced version. We also synchronize and subsequently pseudonymize the changes which have been made in meantime.

9. After deployment we hold control meetings to assure that problems and misunderstandings during the requirements analysis-, design- and implementation phase can be fixed. We repeat the controls and tests until necessary to establish a satisfying solution.

Summary In this section we introduced our proof-of-concept prototype which is the implementation of our novel formal workflow framework. We used the design research approach in combination with the development method rapid prototyping to establish the basis of our feasibility study. The results of the conducted tests look promising. Moreover, the user acceptance of PIPE's additional security layer was high.

Due to our open design it is possible to easily exchange underlying hardware, server or communication systems. Nevertheless, it is important to follow a strict protocol when integrating our approach into an existing system to avoid vulnerabilities.

7 Conclusions

The introduction of the EHR promises massive savings [40, 41, 151] and a better service quality [13, 39, 74, 145] for the patients. Moreover, as modern health care systems still lack standard processes, such a system could also support the definition and execution of e-Health workflows [70, 78].

As highly sensitive data, for example a HIV-infection or an abortion, is stored and handled in nation-wide medical systems, there is the requirement for assuring the patients' privacy to avoid misuse. Hence, several legal acts in the European Union and the United States of America exist, which are the basis for the realization of EHR systems.

Patients have different perceptions of privacy and their participation in such a system, but need to be informed as they are strongly concerned about their privacy. As a matter of fact, it is a vital success factor for any privacy-related system to communicate all actions undertaken on patients' sensitive medical data.

In this thesis we discussed the topics security and privacy. Appropriate security can only be assured if integrated in the design phase. Security can be expressed in terms of confidentiality, integrity, availability and non-repudiation. Privacy is a special case of security and can therefore only be achieved with an implemented security policy. Security and privacy not only rely on technical but also on organizational aspects. Security is a balancing act, because it also influences for example used resources, usability and efficiency of systems. Thus, most business systems need to adapt their workflows to assure appropriate security and privacy.

Several approaches have been proposed to solve the challenge for implementing the EHR (cf. [107, 109, 110, 112, 114]) but these architectures have vulnerabilities regarding their security. Current approaches (i) rely on a centralized patient-medical data list, which could be attacked form in- or outside and (ii) the dependency on a single pseudonym could lead to a profiling attack because an attacker may guess the patient based on her medical history [107, 109, 110, 114]. We worked out several principles with the focus on assuring the confidentiality, integrity, availability and privacy of sensitive patient-related medical data.

Our approach PIPE is a secure and efficient architecture for the combined primary and secondary usage of health-related data based on these principles. Our system assures that the patient is in full control of her data with the maximum of currently gain able security, achieved by applying authorization on encryption [114], in- and outside the system as well as for all communication. Moreover, we use signed messages and database entries to ensure the medical information's integrity.

We introduced a secure fall-back mechanism if a security token, in our case a smart card has been lost, stolen, compromised or just worn out. We defined the administrative role operator which is in charge to hold a backup of the user keys in a backup keystore. Furthermore, we applied a threshold scheme [131] to securely divide the backup keys between the operators to assure inner system's security. In this thesis we outlined the security of this technique and stated the necessary expenses to be able to recover access to the key and consequently to the pseudonymized data. We also showed how human and machine operators can be set in to lower the costs of our fall-back mechanism.

We abstracted our prototype's basic function and stated the necessary administrative and operational workflows. Thus, we showed that sensitive data can be stored in our system with regards to a strong security and privacy. Nevertheless, all information can be retrieved efficiently via a meta-data concept, which is secured by encryption. Moreover, we introduced a novel pseudonyms mechanism, which allows health care provider to access medical data in an emergency.

Finally, we sketched our current prototype, which can be used to build a new EHR PET or work as a basis for existing systems as PAT. Due to our open architecture, the latter is currently only limited to the used programming languages. Though interoperability is not an issue.

7.1 Research Questions Reviewed

In Section 1.3 we outlined the Research Questions of this thesis. In the following we summarize the answers we elaborated.

Research Question A *Regarding pseudonymization as the chosen privacy-enhancing technology: what are the cornerstones and principles of a secure pseudonymization approach?*

In Section 3.9 we stated ten principles which are vital in order to set-up a secure pseudonymization system holding sensitive medical data. First of all, we proposed the separation of identification data and medical information and the concealment of the association between them by the usage of pseudonyms. As these pseudonyms are unique for every patient, health care provider and medical dataset combination, we further reduced the possibility of profiling attacks. Following European laws we assume that the patient is the owner of her data. Hence, she can authorize additional users like other health care providers or relatives and equip them with partial or full access to her data. We stated that the most secure approach to realize authorization is the usage of encryption. Nevertheless, a role-based access control model is still necessary to establish control which cannot be solely realized by encryption and sharing of keys.

Research Question B *How can a secure and efficient measurement for safeguarding the users' keys be provided? What are the occurring costs of such a vital add-on?*

We introduced the role operator. The users inheriting this role in PIPE are in charge of holding a backup version of other users' private keys. To prevent frauds, we applied Shamir's threshold scheme (cf. Section 4.5) to divide this responsibility between n operators. These n operators are a randomly chosen subset of the set of operators $\mathcal{O}$. As at least one of these n operators, which hold a secret share of a user's inner private key $\widehat{K}_A^{-1}$, may not have a working smart card, we introduced an essential availability constraint. Following Shamir only k out of these n operator have to act together in case a particular user's inner private key has to be re-established. To avoid informal arrangements between the operators we also conceal the association between the operators and their assigned patients by the usage of encryption together with organizational security aspects. We outlined the security constraints of our approach in Section 4.5.1 and Section 4.5.2. In order to reduce the necessary processing time and consequently the occurring costs for this type of backup keystore, we discussed the possibility to share the responsibilities between human and machine operators. In Section 4.5.3 we provided the resulting economical model.

Research Question C *What are the workflows of a secure EHR system which is based on pseudonymization? Which administrative and operational workflows besides those, which are necessary to realize the secure backup system requested in Research Question C, are needed?*

In Section 5 we discussed our novel workflow framework for secure pseudonymization of medical data. This framework consists firstly of the basic functions described in Section 5.1 like PIPE's mutual authentication protocol. Secondly, we provided the workflows of adding users (cf. Section 5.2) to our system with both threshold scheme approaches aforementioned (cf. Section 4.5). Moreover, we stated in Section 5.2 how security tokens can be securely replaced in case a user's smart card has been lost, stolen or compromised. Thirdly, we showed in Section 5.3 how medical data can be added, retrieved and updated in PIPE. Furthermore we discussed the different possibilities to permit and revoke access rights based on our layered access model.

Research Question D *Besides the default access to the medical data, how can ad-hoc access in case of a medical emergency be provided?*

In Section 5.4 we discussed the necessity of an ad-hoc access to certain partitions of the anamnesis, diagnosis or treatment data, the emergency data. We introduced a novel authentication technique, which is based on HMAC pseudonyms. We showed that this functionality prevents

replay-attacks and can also be executed on mobile emergency devices. Hence, we assured that PIPE provides emergency doctors with vital information even if the patient is unconscious.

7.2 Contributed Knowledge

The main contribution of this thesis was the creation of a secure formal workflow framework for the pseudonymization of medical data. This novel framework consists of administrative, operational and emergency workflows.

After conducting a secondary literature study as well as workshops for quality assurance, we introduced ten principles for the realization of a secure pseudonymization approach for medical systems. These principles formed the basis for the set-up of our architecture PIPE. Our approach consists of a layered security model in combination with a secure backup keystore based on the elaborated security constraints.

Following the design research approach, we used this architecture to originate the workflow framework. As the access to PIPE is secured by smart cards as security tokens, we showed how the possibility of insider and outsider attacks can be decreased by the appliance of a threshold scheme. To support decision-makers, we introduced the security and economy constraints of our backup approach. Afterwards, we showed how data and right management can be established by using of encryption and sharing keys or hidden relations. We covered administrative and operational workflows in our formal framework. Finally, we discussed our novel secure ad-hoc authentication method, which is also capable of being executed on mobile devices with little memory and a slow CPU.

We evaluated our novel workflow framework throughout the project as follows: We assured that the functionality of our approach does fulfill on the requirements of patients and health care teams by holding regular meetings with our business partners. Moreover, we concurrently developed a prototype and conducted mission load tests in order to measure the additional overhead of PIPE. We used the results of our load tests as input for rapid prototyping in order to improve our system until we gained a satisfying solution. The latest version of this proof-of-concept prototype has been deployed and tested in the network of Genosense Diagnostics.

7.3 Achieved Results and Limitations

As medical data, especially medical images, tend to be very large, we based our system on pseudonymization instead of encryption to save time and costs. This technique assures efficient and secure access to the patients' sensitive data. In addition we worked out a backup approach

to safeguard the patients' private keys. As perfect security usually cannot be assured, we investigated the security constraints of PIPE's backup approach. We implemented a "What-If" analyzes spreadsheet which is based on the formulas of Section 4.5. We further worked out the economical factors of our backup keystore.

To evaluate our approach, we designed and implemented PIPE as an encapsulated additional layer for an existing or a new EHR system. Hence, our approach is applicable as PAT in legacy applications or as new PET architecture. We deployed our proof-of-concept prototype in our business partner's network, in order to get user feedback of an real-world environment and to ease the integration of PIPE for developers. Due to generalization in our formal workflow frameset and subsequently in our prototype, PIPE is not only applicable for medical purposes, but may also cover other usage areas.

In contrast to other approaches, our system PIPE is appropriate for the primary and secondary usage of medical data. Nevertheless, as the information is stored in plaintext to decrease time and costs for processing anamnesis, diagnosis or treatment information, profiling or joining attacks on the patients' data may be possible. Thus, we introduced several metrics and techniques to avoid privacy invasions. Nevertheless, it rests with the administrators and users of the system to decide which information has to be treated as personal-related.

For example the occurrence of rare diseases may increase the risk of an unauthorized patient's re-identification. Hence, for such a case we propose the usage of encryption instead of pseudonymization to assure the patients' privacy. Regarding the secondary usage of a patient's medical history instead of a single anamnesis, diagnosis or treatment dataset, we also recommend the usage of nonces to salt the tags' ciphertexts as described in the next section.

We developed a novel formal workflow framework for which we have been granted an Austrian Patent (cf. [112]). The examination process of our application for a World Patent, which commonly lasts several years, is still in progress (cf. [113]).

We also published different aspects of PIPE on international conferences and improved our system by including the feedback from the research community. In detail we discussed the architecture in [114], the backup keystore in [109] and substantial parts of our formal workflow framework in [107]. Moreover, we introduced our two-folded threshold scheme approach and its security and economic aspects in [108]. We further presented our ad-hoc mechanism for the realization of an emergency access in [111]. In addition we published a concluding Journal Paper about PIPE [110].

Regarding the mid-term development of PIPE, we have been granted a research project by the

Austrian Federal Ministry of Economics and Labor. We introduce the constraints of this project in the next section. Moreover, we are currently realizing the planned integration of PIPE in the laboratories of Genosense Diagnostics.

8 Further Work

In 2008, the Institute of Software Technology and Interactive Systems of the University of Technology in Vienna[25)] together with the Department of Business Informatics - Data & Knowledge Engineering of the University in Linz[26)] and our industry partners, Genosense Diagnostics[27)] and Braincon Technologies[28)] have been granted a research project for further development of PIPE by the Austrian Federal Ministry of Economics and Labor (BMWA).

In particular, we plan (a) to broaden our patented approach to provide for more complex semi-structured or structured meta data than simple tags as described in the patent, (b) to provide alternative secure storage and retrieval services for pseudonymization meta data and selected medical data, (c) to support, beyond smart cards, alternative secure storage and access to medical emergency data, (d) to provide a secure viewing mechanism, and (e) to demonstrate our system in the context of genome analysis, storage and retrieval in the health workflows of our major industrial partner Genosense Diagnostics.

In detail, our plans for the upcoming years are regarding storage and access privacy. We will investigate two approaches: (1) employing block ciphers (where same plaintext values appear with the same cipher value in possible different tuples of the same or different relational tables), which reduce the need to perform query processing at the client, but may open some very limited risk to statistical attacks, and (2) employing stream ciphers or block ciphers with nonces in "stream cipher mode" (where same plaintext values are encrypted differently at each occurrence) together with auxiliary hash index structures (employing cryptographic hash functions) to enable basic search over the encrypted data, which require more encryption/decryption operations (due to the auxiliary hash index structures) but are protected against statistical attacks.

Rather than providing a general purpose data store with arbitrary query functionality, we plan to develop a dedicated database (which may well be built upon a commercial database management system) supporting a predefined set of queries required in our context. This approach allows us to tailor storage and index structures such that a minimum amount of encrypted index structures need to be traversed or processed at the client. The nature of queries and the nature of the data storage is very helpful in our context. Our initial analysis shows that only a few of the expected queries are range queries, requiring that indexes are encrypted/decrypted at the client side for reasons of privacy (i.e. by the smart card or mobile phone if the client is considered un-trusted as well).

[25)] `http://www.ifs.tuwien.ac.at/`
[26)] `http://www.dke.jku.at/`
[27)] `http://www.genosense.at/`
[28)] `http://www.bct.co.at/`

Furthermore, we will be able to partition indexes per patient or health care provider (since data are naturally partitioned and encrypted differently) so that each index partition is rather small and can be quickly traversed. Therefore, we expect that our dedicated solution will be able to deliver acceptable performance, especially if alternative secure access via mobile phones is used for more complex queries.

Regarding to data integrity, we tackle the performance problem in the context of smart cards by relaxing the requirement that a user is able to directly check the authenticity of each query result. We strive rather for checking the authenticity of the overall meta data stored about a patient once the patient initiates an authenticity check. This relaxed requirement is practical in our setting since the meta data about a patient in our system is expected to be of a size that can be realistically retrieved in total for an integrity check. Therefore, we can employ a very simple approach to ensure data integrity.

For example, it will be sufficient to sign every tuple of a relational table to check for modifications and to keep a signed tuple count per patient to check for integrity with respect to additions and deletions. To post-authenticate a specific query result - if it seems doubtful - the patient or operator on her behalf needs to initiate an integrity check over her meta data and compare it against the query result. While such a check may take some time, due to the number of necessary encryption operations at the smart card or mobile phone, we assume it be rare. Alternatively, a patient may carry out a post-authentication more efficiently at a trusted client, e.g. from her personal PC at home.

References

[1] Ackerman, M., Craft, R., Ferrante, F., Kratz, M., Mandil, S., and Sapci, H. Telemedicine Technology. *Telemedicine Journal and e-Health 8* (2002), 71–78.

[2] Adams, T., Budden, M., Hoare, C., and Sanderson, H. Lessons from the central Hampshire electronic health record pilot project: issues of data protection and consent. *BMJ 328* (2004), 871–874.

[3] Agrawal, R., Kiernan, J., Srikant, R., and Xu, Y. Hippocratic Databases. In *Proceedings of 28th International Conference on Very Large Databases* (2002).

[4] Ahn, G., and Sandhu, R. Role-based authorization constraints specification. *ACM Transactions on Information Systems 3*, 4 (2000), 207–226.

[5] Aizcorbe, A., Flamm, K., and Khurshid, A. The role of semiconductor inputs in IT hardware price decline: computers vs. communications. Finance and Economics Discussion Series FEDS Working Paper 2002-37, Board of Governors of the Federal Reserve System (U.S.), 2002.

[6] Anderson, M., and Apap, J. *Striking a Balance between Freedom, Security and Justice in an enlarged European Union.* Centre for European Policy Studies, 2002.

[7] Anderson, R. A security policy model for clinical information systems. In *Institute of Electrical and Electronics Engineers (IEEE) Symposium on Security and Privacy* (1996), pp. 30–43.

[8] Apostolopoulos, G., Peris, V. G. J., and Saha, D. Transport layer security: How much does it really cost? In *Proceedings of Conference on Computer Communications (IEEE Infocom)* (1999), pp. 717–725.

[9] Avizienis, A., Laprie, J., Randell, B., and Landwehr, C. Basic Concepts and Taxonomy of Dependable and Secure Computing. *Transactions on Dependable and Secure Computing 1* (2004), 11–33.

[10] Badder, G., and Winau, R. Generic form in Epidemics I to VII. In *Die Hippokratischen Epidemien: Theorie - Praxis - Tradition* (1989).

[11] Bansal, D., and Goel, A. Interrelation Among RSA Security, Strong Primes, and Factoring. In *Proceedings of Challenges & Opportunities in Information Technology* (2007).

[12] Baron, R., Fabens, E., Schiffman, M., and Wolf, E. Electronic Health Records: Just around the Corner? Or over the Cliff? *Annals of Internal Medicine 143*, 3 (2005), 222–226.

[13] Bates, D. W. Medication errors : How common are they and what can be done to prevent them? *Drug safety 15* (1996), 303–310.

[14] Baumer, D., Earp, J., and Payton, F. Privacy of Medical Records: IT Implications of Health Insurance Portability and Accountability Act (HIPAA). *ACM Computers and Society 30* (2000), 40–47.

[15] Bennett, C. J., and Grant, R. *Visions of Privacy: Policy Choices for the Digital Age*. University of Toronto Press, 1999.

[16] Biryukov, A., Shamir, A., and Wagner, D. Real Time Cryptanalysis of A5/1 on a PC. In *Proceedings of the 7th International Workshop on Fast Software Encryption* (2001), pp. 1–18.

[17] Bishop, M. *Computer Security: Art and Science*. Addison-Wesley (7th Edition), 2005.

[18] Biskup, J., and Flegel, U. Transaction-Based Pseudonyms in Audit Data for Privacy Respecting Intrusion Detection. In *Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection* (2000), pp. 28–48.

[19] Blobel, B., Hoepner, P., Joop, R., Kamouskos, S., Kleinhuis, G., and Stassinopoulos, G. Using a privilege management infrastructure for secure web- based e-health applications. *Computer Communications 16* (2003), 1863–1872.

[20] Blobel, B., and Roger-France, F. A systematic approach for analysis and design of secure health information systems. *International Journal of Medical Informatics 62* (2001), 51–78.

[21] Borcea-Pfitzmann, K., Franz, E., and Pfitzmann, A. Usable presentation of secure pseudonyms. In *Proceedings of the 2005 workshop on Digital identity management* (2005), pp. 70–76.

[22] Borking, J., and Raab, C. Laws, PETs and Other Technologies for Privacy Protection. *Journal of Information, Law and Technology 1* (2001).

[23] Bretherton, F., and Singley, P. Metadata: a user's view. In *Proceedings of the 7th international conference on Scientific and Statistical Database Management* (1994), IEEE Computer Society, pp. 166–174.

[24] California HealthCare Foundation. National Consumer Health Privacy Survey. online, http://www.chcf.org/documents/ihealth/ConsumerPrivacy2005ExecSum.pdf, 2005.

[25] Cannon, J. *Privacy: What Developers and IT Professionals Should Know*. Addison-Wesley Professional, 2004.

[26] Casey, E. *Digital Evidence and Computer Crime*. Second Edition Academic Press, 2004.

[27] Cavoukian, A., and Borking, J. Privacy-enhancing technologies: The path to anonymity. online, http://www.ipc.on.ca/, 1995.

[28] Cavoukian, A., and Hamilton, T. *The Privacy Payoff, How Successful Business Build Consumer Trust*. McGraw-Hill, 2002.

[29] Chhanabhai, P., and Holt, A. EHR Security: The New Zealand Public's Perception. *Health Care and Informatics Review Online* (2006).

[30] Chin, T. Small practices fuel sales of EMR systems. *American Medical News 9* (2004).

[31] Clauss, S., and Pfitzmann, A. Privacy-Enhancing Identity Management. *The Institute for Prospective Technological Studies Report 67* (2002), 8–16.

[32] Codd, E. F. A relational model of data for large shared data banks. *Communications of ACM 26*, 1 (1983), 64–69.

[33] Cooley, T. *Cooley on torts 29*. Harvard Law Review, 1888.

[34] Council of Europe. *European Convention on Human Rights*. Martinus Nijhoff, 1987.

[35] Croll, P. R., and Croll, J. Q.U.i.P.S. – a quality model for investigating risk exposure in e-health systems. *Studies in health technology and informatics 107* (2004), 1023–1027.

[36] Diffie, W., and Hellman, M. New directions in cryptography. *Information Theory, IEEE Transactions on 22*, 6 (1976), 644–654.

[37] Diffie, W., and Hellman, M. E. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *IEEE Computer 10*, 6 (1977), 74–84.

[38] Dyson, F. Time without end: Physics and biology in an open universe. *Reviews of Modern Physics 51*, 3 (Jul 1979), 447–460.

[39] Elson, R. B., and Connelly, D. P. Computerized patient records in primary care. their role in mediating guideline-driven physician behavior change. *Family Medicine 4* (1995), 698–705.

[40] Ernst, F., and Grizzle, A. Drug-Related Morbidity and Mortality: Updating the Cost-of-Illness Model. Tech. rep., Center for Pharmaceutical Economics, College of Pharmacy, University of Arizona, Tucson, USA, 1995.

[41] Ernst, F., and Grizzle, A. Drug-Related Morbidity and Mortality: Updating the Cost-of-Illness Model. Tech. rep., Center for Pharmaceutical Economics, College of Pharmacy, University of Arizona, Tucson, USA, 2001.

[42] EU Council Secretariat. The European Union and the fight against terrorism. http://www.eurunion.org/partner/EUUSTerror/EURespUSTerror.htm, 2007.

[43] European Union. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *Official Journal of the European Communities 11 23, L 281* (1995), 31–50. http://europa.eu/scadplus/leg/en/lvb/l14012.htm.

[44] European Union, Article 29 Working Party. Working document on the processing of personal data relating to health in electronic health records (EHR), February 2007.

[45] European Union, Article 29 Working Party. Opinion 4/2007 on the concept of personal data, June 2007.

[46] Ferraiolo, D., Sandhu, R., Gavrila, S., Kuhn, D., and Chandramouli, R. Proposed NIST standard for role-based access control. *ACM Transactions on Information Systems 4*, 3 (2001), 224–274.

[47] Fischer, L. *Workflow Handbook 2001.* Workflow Management Coalition, 2001.

[48] Flegel, U. Pseudonymizing Unix Log Files. In *Proceedings of the International Conference on Infrastructure Security* (2002), pp. 162–179.

[49] Flinn, B., and Maurer, H. Levels of Anonymity. *Journal of Universal Computer Science 1*, 1 (1995), 35–47.

[50] Ganley, M. Elliptic Curve Cryptography - an Introduction. In *Proceedings of Information Security Bulletin* (2001), pp. 25–28.

[51] Gavish, B., and Gerdes, Jr., J. Anonymous mechanisms in group decision support systems communication. *Decision Support Systems 23*, 4 (1998), 297–328.

[52] Givens, P. Medical records privacy: fears and expectations of patients. In *Proceedings of Toward an Electronic Patient Record Conference* (2005).

[53] Gollmann, D. *Computer Security.* John Wiley & Sons, 1999.

[54] Grimson, J. Delivering the electronic healthcare record for the 21st century. *International Journal of Medical Informatics 64* (2001), 111–127.

[55] Grochowski, E., and Halem, R. D. Technological impact of magnetic hard disk drives on storage systems. *IBM Systems Journal* (2003).

[56] Han, J., Kim, Y., Jun, S., Chung, K., and Seo, C. Implementation of ECC/ECDSA Cryptography Algorithms Based on Java Card. In *Proceedings of the 22nd International Conference on Distributed Computing Systems* (2002), pp. 272–278.

[57] Hankerson, D., Menezes, A. J., and Vanstone, S. *Guide to Elliptic Curve Cryptography*. Springer New York, 2003.

[58] He, Q., and Antn, A. A Framework for Modeling Privacy Requirements in Role Engineering. In *Proceedings of the Workshop on Requirements Engineering for Software Quality* (2003).

[59] Hendry, M. *Smart Card Security and Applications, Second Edition*. Artech House, Inc., 2001.

[60] IBM. Machbarkeitsstudie betreffend Einfuehrung der elektronischen Gesundheitsakte (ELGA) im oesterreichischen Gesundheitswesen, 2006.

[61] Jiang, X., and Landay, J. Modeling Privacy Control in Context-Aware Systems. *IEEE Pervasive Computing 1*, 3 (2002), 59–63.

[62] Johnson, D., and Menezes, A. The Elliptic Curve Digital Signature Algorithm (ECDSA). Tech. rep., University of Waterloo, 1999.

[63] Jorns, O., Jung, O., and Quirchmayr, G. Transaction Pseudonyms in Mobile Environments. *Journal in Computer Virology 3* (2007), 185–194.

[64] Kahn, D. *The Code Breakers: The Story of Secret Writing*. Macmillan, 1967.

[65] Kerckhoffs, A. La cryptographie militaire. *Journal Sciences Militaires 9* (1883), 5–38.

[66] Koblitz, N. Elliptic Curve Cryptosystems. *Mathematics of Computation 48* (1987), 203–209.

[67] Kohn, L. T., Corrigan, J., and Donaldson, M. S. *To Err Is Human: Building a Safer Health System*. National Academies Press, 2000.

[68] Kuhn, K., and Giuse, D. From hospital information systems to health information systems. problems, challenges, perspectives. *Methods of informations in medicine 40* (2001), 275–287.

[69] LaPadula, L., and Bell, D. MITRE technical report 2547, volume II. *Journal of Computer Security 4*, 2-3 (1996), 239–263.

[70] Leape, L. L., and Berwick, D. M. Five Years After To Err Is Human - What Have We Learned? *Journal of the American Medical Association 293* (2005), 2384–2390.

[71] LeFevre, K., DeWitt, D. J., and Ramakrishnan, R. Mondrian Multidimensional K-Anonymity. In *Proceedings of the 22nd International Conference on Data Engineering* (2006).

[72] Liu, C. *Introduction to Combinatoral Mathematics*. McGraw-Hill, New York, 1968.

[73] Lysyanskaya, A., Rivest, R., Sahai, A., and Wolf, S. Pseudonym Systems. In *Proceedings of the Sixth Annual Workshop on Selected Areas in Cryptography* (1999).

[74] Maerkle, S., Koechy, K., Tschirley, R., and Lemke, H. The PREPaRe system – Patient Oriented Access to the Personal Electronic Medical Record. In *Proceedings of Computer Assisted Radiology and Surgery, Netherlands* (2001), pp. 849–854.

[75] Marcella, A., and Carol, S. *Privacy Handbook: Guidelines, Exposures, Policy Implementation, and International Issues*. Wiley, 2003.

[76] Marco, D. *Building and managing the Meta Data Repository: A Full Life-Cycle Guide*. John Wiley & Sons, Inc., 2000.

[77] Maris, K. The Human Factor. In *Proceedings of Hack.lu* (2005).

[78] McGlynn, E. A., Asch, S. M., Adams, J., Keesey, J., Hicks, J., DeCristofaro, A., and Kerr, E. A. The Quality of Health Care Delivered to Adults in the United States. *The New England Journal of Medicine 348* (2003), 2635–2645.

[79] Menezes, A., van Oorschot, P., and Vanstone, S. *The Handbook of Applied Cryptography*. CRC Press, 1997.

[80] Miller, R., Sim, I., and Newman, J. Electronic medical records: lessons from small physician practices. Tech. rep., 2004.

[81] Miller, V. S. Use of Elliptic Curves in Cryptography. In *Proceedings of Advances in Cryptology* (1986), pp. 417–426.

[82] Mollin, R. *Codes: The Guide to Secrecy from Ancient to Modern Times*. CRC Press, 2005.

[83] Monger, J. International comparisons of labour disputes in 2002. *Labour Market Trends 111* (2003), 19–28.

[84] Montagnat, J., Bellet, F., Benoit-Cattin, H., Breton, V., Brunie, L., Duque, H., Legr, Y., Magnin, I., Maigne, L., Miguet, S., Pierson, J., Seitz, L., and Tweed, T. Medical Images Simulation, Storage, and Processing on the European DataGrid Testbed. *Journal of Grid Computing 2* (2004), 387–400.

[85] National Institute of Standards and Technology. Advanced Encryption Standard. In *Federal Information Processing Standards (FIPS) 197* (2001).

[86] National Institute of Standards and Technology. Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA3) Family. *Federal Register Vol. 72, No. 212* (2002), 62212–62220.

[87] National Institute of Standards and Technology. Secure hash standard (SHS). In *Federal Information Processing Standards Publication 180-2* (2002).

[88] Needham, R. M., and Schroeder, M. D. Using encryption for authentication in large networks of computers. *Communications of ACM 21*, 12 (1978), 993–999.

[89] Network of Excellence in Cryptology. Recent Collision Attacks on Hash Functions. In *ECRYPT Position Paper* (2005).

[90] Nielsen, J. *Usability engineering*. Morgan Kaufmann, 1993.

[91] Nutbeam, D. Health promotion glossary. *Health Promotion International 13* (1998), 349.

[92] Office for National Statistics (ONS). Average total income and average income tax payable: by sex, 2000/01: Regional Trends 38. ONLINE, 2002. http://www.statistics.gov.uk/StatBase/ssdataset.asp?vlnk=7752.

[93] Office for National Statistics (ONS). T 04: England; estimated resident population by single year of age and sex; Mid-2005 Population Estimates. ONLINE, 08 2006. http://www.statistics.gov.uk/statbase/Product.asp?vlnk=14508&More=Y.

[94] Onuma, L., Ito, A., Mossman, B., Albertson, R., and Hilton, S. Patent: System and Method for Controlling Access and Use of Patient Medical Data Records. *US Patent US 2005/0236474 A1* (2005).

[95] Otway, D., and Rees, O. Efficient and timely mutual authentication. *SIGOPS Operating Systems Review 21*, 1 (1987), 8–10.

[96] Peterson, R. Patent: Encryption System for allowing immediate universal access to medical records while maintaining complete patient control over privacy. *US Patent US 2003/0074564 A1* (2003).

[97] Pfitzmann, A., and Koehntopp, M. Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity Management A Consolidated Proposal for Terminology. In *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2005.

[98] Pommerening, K. Medical Requirements for Data Protection. In *Proceedings of IFIP Congress, vol. 2* (1994), pp. 533–540.

[99] Pommerening, K., and Reng, M. Secondary use of the Electronic Health Record via pseudonymisation. *Medical Care Compunetics 1* (2004), 441–446.

[100] Pope, J. Implementing EHRs requires a shift in thinking. PHRs–the building blocks of EHRs–may be the quickest path to the fulfillment of disease management. *Health Management Technology 27* (2006), 24,26,120.

[101] Popek, G. A principle of kernel design. In *Proceedings of AFIPS Conference* (1974).

[102] Privacy Rights Clearinghouse. Fact Sheet 8(a): HIPAA Basics: Medical Privacy. In *UCAN* (2007).

[103] Rankl, W., and Effing, W. *Smart Card Handbook*. John Wiley & Sons, Inc., 1997.

[104] Rector, A., Rogers, J., Taweel, A., Ingram, D., Kalra, D., Milan, J., Singleton, P., Gaizauskas, R., Hepple, M., Scott, D., and Power, R. CLEF - Joining up Healthcare with Clinical and Post-Genomic Research. In *Proceedings of UK e-Science All Hands Meeting* (2003), Healthcare Computing 2004, pp. 203–211.

[105] Reeves, C. A., and Bednar, D. A. Defining Quality: Alternatives and Implications. *The Academy of Management Review 19* (1994), 419–445.

[106] Republic of Austria. Datenschutzgesetz 2000 (DSG 2000), BGBl. I Nr. 165/1999, 1999.

[107] Riedl, B., Grascher, V., Fenz, S., and Neubauer, T. Pseudonymization for improving the Privacy in e-Health Applications. In *Proceedings of the Forty-First Hawai'i International Conference on System Sciences* (2008), p. 255.

[108] Riedl, B., Grascher, V., Kolb, M., and Neubauer, T. Economic and Security Aspects of the Appliance of a Threshold Scheme in e-Health. In *Proceedings of the Third International Conference on Availability, Reliability and Security* (2008), pp. 39–46.

[109] Riedl, B., Grascher, V., and Neubauer, T. Applying a Threshold Scheme to the Pseudonymization of Health Data. In *Proceedings of the 13th IEEE Pacific Rim International Symposium on Dependable Computing* (2007), pp. 397–400.

[110] Riedl, B., Grascher, V., and Neubauer, T. A Secure e-Health Architecture based on the Appliance of Pseudonymization. *Journal of Software 3* (2008), 23–32.

[111] Riedl, B., and Jorns, O. Secure Access to Emergency Data in an e-Health Architecture. In *Proceedings of the 9th International Conference on Information Integration and Web-based Application & Services* (2007), pp. 297–306.

[112] Riedl, B., Neubauer, T., and Boehm, O. Datenverarbeitungssystem zum Verarbeiten von Objektdaten, 2006. Austrian-Patent, No. A 503 291 B1, 2007-09-15.

[113] Riedl, B., Neubauer, T., and Boehm, O. System for Processing of Object Data, 2007. Application No. WO/2008/061267.

[114] Riedl, B., Neubauer, T., Goluch, G., Boehm, O., Reinauer, G., and Krumboeck, A. A secure architecture for the pseudonymization of medical data. In *Proceedings of the Second International Conference on Availability, Reliability and Security* (2007), pp. 318–324.

[115] Rijmen, V. The First 10 Years of Advanced Encryption. In *Speech at Secure Business Austria, Vienna* (2007).

[116] Rindfleisch, T. C. Privacy, information technology, and health care. *Communications of ACM 40*, 8 (1997), 92–100.

[117] Rivest, R. L., Shamir, A., and Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Communications of ACM 21*, 2 (1978), 120–126.

[118] Rolfe, J. *The Loeb Classical Library*, vol. 2. Harvard University Press, Cambridge, 1914, ch. Suetonius: The Lives of the Caesars.

[119] Russell, R., Kaminsky, D., Puppy, R. F., Grand, J., Ahmad, D., Flynn, H., Dubrawsky, I., Manzuik, S. W., and Permeh, R. *Hack Proofing Your Network (Second Edition)*. Syngress Publishing, 2002.

[120] Saltzer, J., and Schroeder, M. The Protection of Information in Computer Systems. *Communications of the ACM 17* (1974).

[121] Samarati, P., and Sweeney, L. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Tech. rep., SRI Computer Science Laboratory, 1998.

[122] Sandhu, R., and Samarati, P. Authentication, access control, and audit. *ACM Computing Survey 28*, 1 (1996), 241–243.

[123] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. Role-Based Access Control Models. *IEEE Computer 29* (1996), 38–47.

[124] Sandhu, R. S., and Samarati, P. Access Control: Principles and Practice. *IEEE Communications Magazine 32*, 9 (1994), 40–48.

[125] Schartner, P., and Schaffer, M. Unique User-Generated Digital Pseudonyms. *Computer Network Security 3685* (2005), 194–205.

[126] Schmidt, V., Striebel, W., Prihoda, H., Becker, M., and Lijzer, G. D. Patent: Verfahren zum Be- oder Verarbeiten von Daten. *German Patent, DE 199 25 910 A1* (2001).

[127] Schneider, F. B. Least Privilege and More. *IEEE Security and Privacy 1*, 5 (2003), 55–59.

[128] Schneier, B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* Wiley; 2. Edition, 1995.

[129] Schreck, J. *Security and Privacy in User Modeling.* Kluwer Academic Publishers, 2003.

[130] Seničar, V., Jerman-Blažič, B., and Klobučar, T. Privacy-Enhancing Technologies: approaches and development. *Computer Standards & Interfaces 25*, 2 (2003), 147–158.

[131] Shamir, A. How to share a secret. *Communications of ACM 22*, 11 (1979), 612–613.

[132] Silverman, R. D. Exposing the Mythical MIPS Years. *IEEE Computer 32* (1999), 22–26.

[133] Slamanig, D., and Stingl, C. Privacy Aspects of eHealth. In *Proceedings of the Third International Conference on Availability, Reliability and Security* (2008), pp. 1226–1233.

[134] Stewart, A. L., and Ware, J. E. *Measuring Functioning and Well-Being: The Medical Outcomes Study.* Duke University Press, 1992.

[135] Sweeney, L. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10* (2002), 557–570.

[136] Sweeney, L. k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10* (2002), 571–588.

[137] Swire, P., and Steinfeld, L. Security and privacy after September 11: the health care example. In *Proceedings of the 12th annual conference on Computers, freedom and privacy* (2002), pp. 1–13.

[138] T. Tavani, H., and Moor, J. Privacy protection, control of information, and privacy-enhancing technologies. *Special Interest Group on Computers and Society 31*, 1 (2001), 6–11.

[139] Taipale, K. Technology, Security and Privacy: The Fear of Frankenstein, the Mythology of Privacy and the Lessons of King Ludd. *International Journal of Communications Law & Policy 9* (2004).

[140] The Center for Information Technology Leadership. The value of healthcare information exchange and interoperability. *Healthcare Information and Management Systems Society* (2005).

[141] Thielscher, C., Gottfried, M., Umbreit, S., Boegner, F., Haack, J., and Schroeders, N. Patent: Data processing system for patient data. *International Patent, WO 03/034294 A2* (2005).

[142] Thomas, B., Quirchmayr, G., and Piekarski, W. Through-Walls Communication for Medical Emergency Services. *International Journal of Human-Computer Interaction 16* (2003), 477–496.

[143] Thomson, D., Bzdel, L., Golden-Biddle, K., Reay, T., and Estabrooks, C. Central Questions of Anonymization: A Case Study of Secondary Use of Qualitative Data. *Forum Qualitative Social Research 6* (2005).

[144] Thornburgh, T. Social engineering: the "Dark Art". In *Proceedings of the 1st annual conference on Information security curriculum development* (2004), pp. 133–135.

[145] Trivedi, M. H., Kern, J. K., Grannemann, B. D., Altshuler, K. Z., and Sunderajan, P. A Computerized Clinical Decision Support System as a Means of Implementing Depression Guidelines. *Psychiatric Services 55* (2004), 879–885.

[146] United Nations. Universal Declaration of Human Rights. In *General Assembly resolution 217 A (III)* (1948).

[147] United States Department of Health & Human Service. Health Insurance Portability and Accountability Act (HIPAA) Administrative Simplification: Enforcement; Final Rule. *Federal Register / Rules and Regulations 71* (2006).

[148] van Bemmel, J., and Musen, M. *Handbook of Medical Informatics*. Springer, 1997.

[149] W3C. Soap version 1.2. In *W3C Working Draft* (2001).

[150] Wahlstrom, K., and Quirchmayr, G. The motivation and proposition of a privacy-enhancing architecture for operational databases. In *Proceedings of the fifth Australasian symposium on ACSW frontiers* (2007), pp. 173–182.

[151] Wang, S., Middleton, B., Prosser, L., Bardon, C., Spurr, C., Carchidi, P., Kittler, A., Goldszer, R., Fairchild, D., Sussman, A., Kuperman, G., and Bates, D. A cost-benefit analysis of electronic medical records in primary care. *The American Journal of Medicine 114* (2003), 397–403.

[152] Weippl, E., and Riedl, B. *Handbook of research on Mobile Multimedia: 2nd edition*. 2007, ch. Security, Trust and Privacy on Mobile Devices and Multimedia Applications.

[153] Westin, A. *Privacy and Freedom*. Atheneum, 1968.

[154] Westran, T., Mack, M., and Enbody, R. The Last Line of Defense: a Host-Based, Real-Time, Kernel-Level Intrusion Detection System. Tech. Rep. MSU-CSE-05-16, Department of Computer Science, Michigan State University, 2005.

[155] WHO. Constitution of the World Health Organization. In *Geneva* (1946).

[156] WHO. The move towards a new public health. In *Ottawa Charter for Health Promotion, First International Conference on Health Promotion* (1986).

[157] WHO. Global strategy on occupational health for all: The way to health at work. In *Recommendation of the second meeting of the WHO Collaborating Centres in Occupational Health, Beijing* (1994).

[158] Winkler, D., Riedl, B., and Biffl, S. Improvement of Design Specifications with Inspection and Testing. In *Proceedings of the 31st EUROMICRO Conference on Software Engineering and Advanced Applications* (2005), pp. 222–231.

[159] Wu, T. The Secure Remote Password Protocol. In *Proceedings of the Internet Society Network and Distributed System Security Symposium* (1998), pp. 97–111.

[160] Yoder, J., and Barcalow, J. Architectural patterns for enabling application security. In *Proceedings of the 4th Conference on Patterns Language of Programming* (1997).

[161] Zhang, L., Ahn, G., and Chu, B. A role-based delegation framework for healthcare information systems. In *Proceedings of the seventh ACM symposium on Access control models and technologies* (2002), pp. 125–134.

List of Figures

List of Tables

Appendix - Prototype Database Structure

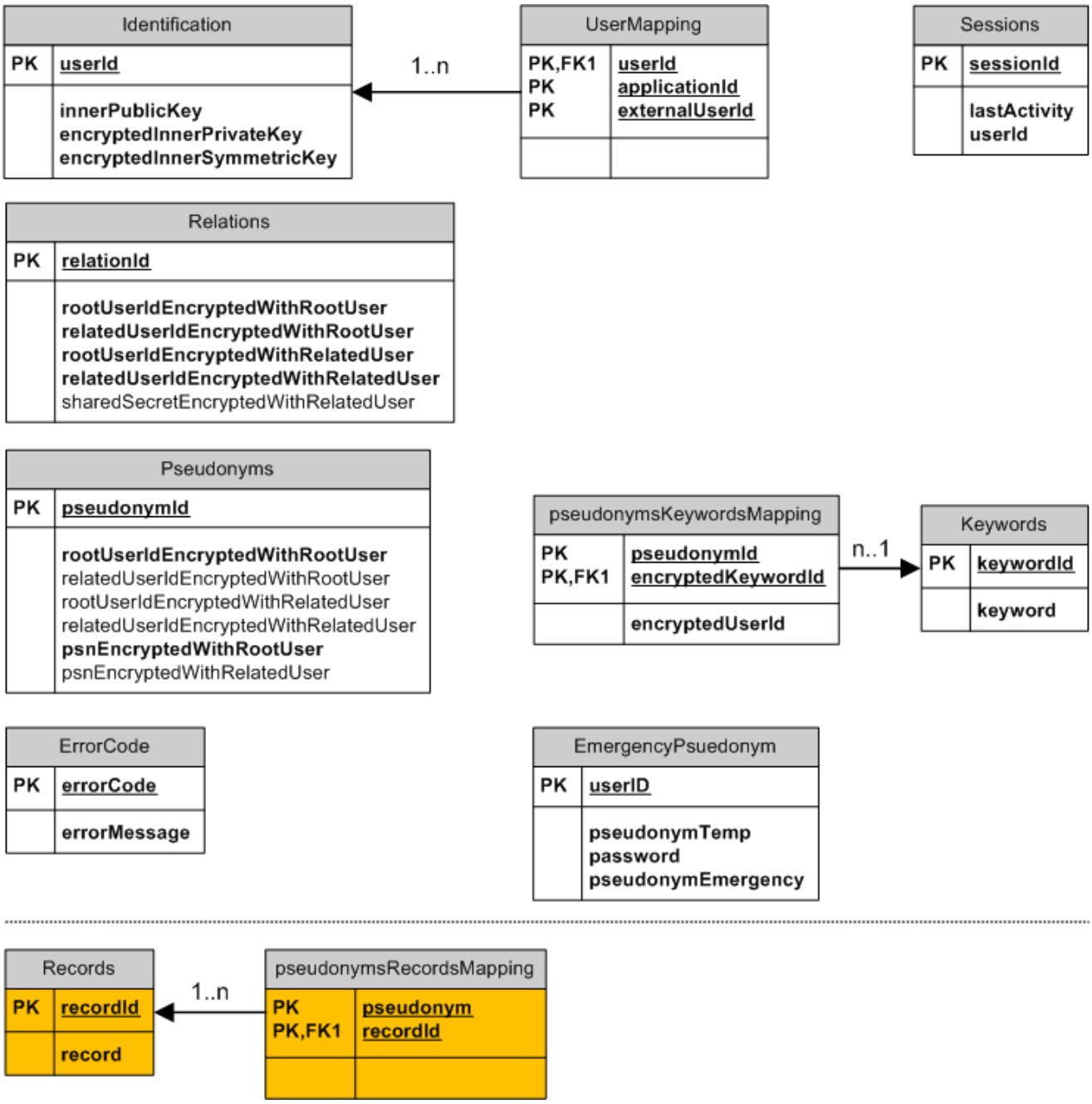

Figure 22: Database Diagram of PIPE's Proof-of-Concept Prototype Logic Module

Appendix - Prototype Function List

Global API:

- addData
- addDataRelation
- addKeyword
- getKeywords
- getKeywordIDs
- retrieveData
- revokeDataRelation
- searchData
- isRootUser
- getRelatedUser
- showErrorGUI
- addRelationAssociate
- addRelationAuthorize
- updateData
- updateDataKeyword

Internal API:

- authenticateUser
- createSession
- lookUpUser

- lookUpUserForAuthentication
- readSession
- decryptWithInnerPrivateKey
- decryptWithInnerSymmetricKey
- doDecryptionWithInnerPrivateKey
- doDecryptionWithInnerSymmetricKey
- doEncryptionWithInnerPublicKey
- doEncryptionWithInnerSymmetricKey
- encryptWithInnerPublicKey
- encryptWithInnerSymmetricKey
- getInnerPrivateKey
- getInnerPublicKey
- getInnerSymmetricKey
- getInternalUserId

Global Logic:

- addActor
- addData
- addDataRelation
- addKeyword
- authenticateUser
- getErrorMessage
- getKeywords

- getKeywordIDs
- lookUpUser
- retrieveData
- revokeDataRelation
- searchData
- sessionExists
- storePseudonym
- getInnerPrivateKey
- getInnerPublicKey
- getInnerSymmetricKey
- getUserPublicKey
- mapInternalUser
- getRelatedPseudonyms
- isRootPseudonym
- storeActorData

Internal Logic:

- addRelation
- bindActor
- updateData
- updateDataKeyword
- createPseudonym
- pseudonymExist

- createSession
- createInternalUser
- userExists
- encryptWithInnerPublicKey

Global Admin:

- addActor
- bindActor

Internal Admin:

- createCard
- encryptWithInnerPublicKey
- encryptWithOuterPublicKey
- generateAsymmetricKeyPair
- generateSymmetricKey

Acknowledgment

I want to thank

A Min Tjoa and Gerald Quirchmayr

for their guidance with this thesis,

Markus Klemen und Edgar Weippl as representatives for

all members of the competence center Secure Business Austria (SBA)

for giving me the opportunity to conduct research at SBA, which has been partially funded by the Austrian Federal Ministry of Economics and Labor (BMWA) as well as by the provincial government of Vienna,

Davul Ljuhar as representative for

all members of our business partner Braincon Technologies,

for their ideas and discussions regarding PIPE,

Manfred Müller and Christian Schneeberger as representatives for

all members of our business partner Genosense Diagnostics,

for their prototypical implementation of PIPE,

Thomas Neubauer and Manfred Linnert

for their help in the theoretical and practical part of this thesis,

Mathias Kolb and Markus Pehaim

for their participation in the design, development and testing of our prototypes,

Angus Hain as representative for

the inhabitants and travellers of Koh Phangan, Thailand

for their goodwill and motivation,

Veronika Grascher

for her companionship, support and love in good and dark hours while writing

and my parents

Anni & Robert Riedl

for my humble existence.