FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

A framework for the application of pseudonymization for primary and secondary use of health data

DISSERTATION

zur Erlangung des akademischen Grades

Doktor der Sozial- und Wirtschaftswissenschaften

eingereicht von

Mag. Johannes Heurix
Matrikelnummer 0026079

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Univ.Prof. Dipl.Ing. Dr. A Min Tjoa

Diese Dissertation haben begutachtet:

A Min Tjoa	Gerald Quirchmayr

Wien, 15. April 2016

Johannes Heurix

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

A framework for the application of pseudonymization for primary and secondary use of health data

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der Sozial- und Wirtschaftswissenschaften

by

Mag. Johannes Heurix
Registration Number 0026079

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.Prof. Dipl.Ing. Dr. A Min Tjoa

The dissertation has been reviewed by:

A Min Tjoa	Gerald Quirchmayr

Vienna, 15th April, 2016

Johannes Heurix

Erklärung zur Verfassung der Arbeit

Mag. Johannes Heurix
Koppstraße 25/13, 1160 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 15. April 2016

Johannes Heurix

Acknowledgements

First of all, I would like to thank my supervisor Prof. A Min Tjoa for his excellent support and the opportunity to follow my own research ideas and interests. I am also grateful to my colleagues Stefan Fenz and Thomas Neubauer, as well as the colleagues from JKU Linz and Xitrust Secure Technologies, for their valuable input and collaboration in the research projects that were the foundation of this thesis. Furthermore, I would like to thank SBA Research for the opportunity to work on my thesis during the last years. And finally, I would like to thank my family and friends for their support during all those years.

This thesis would not have been possible without the financial support in the form of grants of the Austrian government's FIT-IT (contract 816158) and BRIDGE (contract 824884) research initiatives, as well as COMET K1 (contract 843274) by the FFG Austrian Research Agency.

Kurzfassung

Die heutige Welt ist von der Verfügbarkeit großer Datenmengen und den Technologien, mit Hilfe derer diese Daten verarbeitet werden, geprägt. Diese Entwicklung hat zwar die Wirtschaft merklich angekurbelt, gleichzeitig aber auch die Notwendigkeit für Datensicherheit erhöht. Sobald sensible und persönliche Daten verarbeitet werden, müssen entsprechende Datensicherheitsmechanismen eingesetzt werden, um unbefugte Datenweitergabe zu verhindern, die sich negativ auf Einzelpersonen auswirken könnte. Bei persönlichen Gesundheitsdaten handelt es sich um hochsensible Daten, deren Weitergabe streng kontrolliert werden muss, um die Privatsphäre der betroffenen Personen zu wahren. Die Einführung vernetzter Systeme wie beispielsweise elektronischer Gesundheitsakten hat den Zugriff auf und die Verarbeitung von entscheidenden Informationen erleichtert, was auch zu einer Verbesserung der allgemeinen Gesundheitsversorgung geführt hat. Mit der Vereinfachung des Zugriffs auf kritische Daten ist jedoch auch die Angst vor Datenmissbrauch durch Unbefugte gewachsen. Die unkontrollierte Weitergabe persönlicher Gesundheitsdaten führt in vielen Fällen zu Diskriminierung und Belästigung der betroffenen Personen. Daher sollten die bestehenden Rechtsvorschriften durch technische Maßnahmen ergänzt werden. Da persönliche Gesundheitsdaten aber auch eine wertvolle Informationsquelle für Forschungszwecke sind und Patienten meist mit der Weitergabe ihrer Daten für die Sekundärnutzung durch Dritte einverstanden sind, solange ihre Privatsphäre dabei gewahrt wird, muss das Gleichgewicht zwischen der Wahrung der Privatsphäre der Patienten und der Nutzbarkeit ihrer Daten für die Forschung gefunden werden.

Diese Dissertation untersucht Pseudonymisierung als eine Methode, um dieses Gleichgewicht zwischen Privatsphäre der Patienten und Nutzbarkeit der Daten zu wahren. Die auf Pseudonymisierung basierende Sicherheitsarchitektur gewährleistet, dass die Gesundheitsdaten der Patienten pseudonymisiert gespeichert werden, was die Sekundärnutzung der Daten ermöglicht, während die Privatsphäre der Patienten gewahrt wird. Da Pseudonymisierung ein umkehrbarer Prozess ist, können vertrauenswürdige Gesundheitsdienstleister für die primäre Gesundheitsversorgung Zugang zu den nicht-pseudonymisierten Originaldaten erhalten. Diese Form des autorisierten Datenzugriffs wird ausschließlich von den Patienten als Dateneigentümern gesteuert. Demnach unterstützt die Pseudonymisierungsarchitektur sowohl die vom Patienten gesteuerte Primärdatennutzung sowie die Sekundärdatennutzung unter Wahrung der Privatsphäre der Patienten. Darüber hinaus untersucht die Dissertation auch die Pseudonymisierung für die alleinige Sekundärdatennutzung einschließlich der damit verbundenen notwendigen Schritte für die Umwandlung

von bestehenden archivierten Gesundheitsdaten in eine für die Sekundärdatennutzung für Forschungszwecke geeignete Form, während wiederum die Privatsphäre der Patienten gewahrt wird.

Abstract

Today's world is characterized by the availability of large amounts of data and the technologies to process them. This has been a significant boost to today's economy, but has also increased the need for data security. Whenever sensitive and personal data is involved, adequate data protection mechanisms must be installed to prevent unauthorized data disclosure which results in adverse consequences for individuals. Personal health data is a particular, usually highly sensitive type of data, which is why its disclosure must be tightly controlled in order to protect the privacy of individuals. The introduction of interconnected systems like electronic health records has made it easier to acquire and process vital information and has thus improved general health care, though the facilitated access to critical data has also increased the fear of data abuse by unauthorized parties. More often than not, unregulated disclosure of personal health data leads to discrimination or harassment of the affected individuals. Thus, existing legal regulations should be supplemented by technical means. However, personal health data is also an important source of information for research purposes, and patients usually agree to this form of beneficial data disclosure to third parties for secondary use, as long as their privacy is preserved. Thus, it is necessary to keep the balance between the patients' privacy and the usability of their health data for research purposes.

In this thesis, pseudonymization is investigated as a method to keep this balance between privacy and data usability. The security architecture based on pseudonymization ensures that the patients' health data is stored in a pseudonymized state, which enables privacy-preserving secondary use. Since pseudonymization is a reversible process, access to the original de-pseudonymized data can be granted to trusted health care providers for direct primary care. This form of authorized data access is controlled exclusively by the patients who are acting as owners of their data. Therefore, this pseudonymization architecture supports the concurrent patient-controlled primary use and privacy-preserving secondary use of health data. Furthermore, the thesis also investigates pseudonymization in a scenario purely for secondary use including the necessary steps to convert existing archived health data into a form suitable for privacy-preserving processing for research purposes.

Contents

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction

1.1 Motivation

Data security is one of the key words in today's data centric world. Each day, enormous amounts of sensitive data are created and thus must be protected adequately. Large scale data intrusion incidents such as the Heartland or the Sony incidents [53] clearly demonstrate the need for adequate data security. In 2014, Sony once again was the target of an attack which was even more dramatic than the 2011 incident as it demonstrated the existence of critical vulnerabilities and bad practices such as the existence of folders actually named 'passwords'. The stolen data included sensitive internal documents such as unreleased scripts, but also very personal employee-specific data such as information about the medical conditions of employees and their family members [96]. Financial institutions are also high-profile targets for hackers, as demonstrated by the attack on JPMorgan Chase, also in 2014 [104]. In data security, the notion of privacy is usually associated with confidentiality of data. But in a broader sense, information privacy is defined as "the claim of individuals, groups or institutions to determine for themselves when, how, and to what extend information about them is communicated to others" [124], or in other words, to whom sensitive data is disclosed.

Unregulated disclosure of sensitive information such as credit card information, bank accounts or user profiles of social networks may result in considerable adverse effects for the persons involved when exploited by criminal parties. Furthermore, in times where even the average non-criminal citizen is under surveillance by governments and their intelligence services using clandestine surveillance programs or more overt approaches such as data retention, the need for privacy of individuals is clearly evident. Thus, although privacy is declared as a human right in article 8 of the European Convention on Human Rights [34] and the processing or transfer of personal data is also legally regulated by the European Union with Directive 95/46/EC [44], there is a clear need for supplementing these legal acts with robust technical solutions. These technical solutions are necessary because legal regulations can be amended or overridden when deemed

necessary with severe consequences as demonstrated in the reduction of civil rights by the introduction of the USA Patriot Act [114]. The introduction of more or less legally compliant practices such as the before mentioned unconditional data retention and other surveillance operations has proven that legal regulations are less effective than expected and that they were often plainly ignored by governmental institutions.

Apart from everyday life, a particular domain where privacy plays a large role is health care. The introduction of information and communication technologies into health care has considerably increased the quality of health services. The ability of processing or acquiring the necessary critical information is vital for health practitioners to make the correct decisions concerning treatment plans and therefore saving patients' lives. Interconnected systems like electronic health records (EHRs) were introduced to create a common standard for data storage and especially data exchange of critical information and therefore have the potential to further improve the quality of patient treatment. The Austrian ELGA (Elektronische Gesundheitsakte) [42] initiative and the German EGK (Elektronische Gesundheitskarte) [27] [49] both aim at facilitating the exchange of specific patient-related data between health care providers for that purpose. However, the large amounts of data produced by hospital information systems increase the chance of data abuse by unauthorized parties which might lead to discrimination or harassment: Organizations like insurance companies or employers are highly interested in gathering health-related information on potential customers or employees, and the disclosure of information such as a history of substance abuse or an HIV infection to health insurers or employers could result in denied health insurance coverage or denied job offers. The introduction of the ELGA and the EGK has created the fear of becoming a 'transparent patient' in many persons who are afraid of an unregulated and uncontrolled exchange of sensitive health information. This fear is very well understandable given the numerous documented cases of health information abuse. Furthermore, people do not necessarily need to suffer from a particular illness to experience discrimination. Even the pure predisposition of a certain disease could be sufficient to lead to discriminating situations. This form of genetic discrimination [75] was so wide-spread in the United States that the US government had to react by introducing the Genetic Information Nondiscrimination Act (GINA) which was signed in 2008 [115].

Thus, the discussion of privacy is actually one of the fundamental issues in health care today. The disclosure of health-related data is therefore subject of several data protection acts such as the GINA or the Health Insurance Portability and Accountability Act (HIPAA) [116] issued by the United States Department of Health and Human Services in 1996. Furthermore, technical solutions such as data encryption, which is very common in areas like the banking sector, are also increasingly applied in health care data management to protect sensitive data.

1.2 Problem statement

As beneficial as legal acts such as the HIPAA are in general, these regulations are counterproductive when it comes to beneficial disclosure of health information in the

form of secondary use [78]. Advances in medical techniques are largely the results of empirical studies where the researchers require access to extensive and accurate medical data of patients to back their data-driven research methods. Access to patient data is also necessary when trying to identify possible candidates for clinical studies, which is very time-consuming and thus costly [25]. In general, patients do often agree to make their medical data available and participate in clinical studies, but they express concerns that personal information could be used for marketing and insurance purposes [126], that it could be disclosed to employers and family members [3], or that it could be disclosed due to common security incidents [105]. Thus, it must be ensured that the patients' privacy is preserved even in secondary use, which requires a trade-off between the patients' requirements for privacy and the society's need for improving overall health care.

A technique often mentioned to solve this problem is anonymization during which all identifying data elements are removed in order to make it impossible to reidentify the corresponding individual. In such a scenario, health data would be anonymized before being disclosed to researchers. However, there are several drawbacks with this approach: Anonymization always leads to loss of data accuracy and thus data expressiveness. For example, apart from removing direct identifiers such as names, the well-known anonymization approach of k-anonymity [108] applies generalization and suppression techniques on person-identifying (quasi)identifiers (i.e. elements that are not identifying on their own but are identifying when combined, like the home address or age) to divide patients into equivalence groups including individuals that share the same quasi-identifiers and thus cannot be distinguished from each other. Furthermore, anonymization is an irreversible process and, thus, the original data cannot be restored without access to the original data source, which can be problematic when patients need to be reidentified for, e.g., the selection process of a clinical trial.

A similar approach to anonymization is pseudonymization where the identifying data elements are not completely removed but replaced by pseudonyms. These pseudonyms allow for reidentification of the patients under strictly controlled circumstances. This reversibility effectively circumvents the limitations of anonymization, which renders pseudonymization a superior approach compared to anonymization in this case. Several pseudonymization approaches have been proposed (cf. Section 2.4), but they regard pseudonymization as a mechanism to solely create unlinked medical data records, mainly for secondary use. Other security properties such as access control are usually regarded as being out-of-scope and handled externally by a security infrastructure with separate authentication and authorization controls.

1.3 Research question

This thesis investigates how pseudonymization can be applied to meet general security requirements such as authentication and authorization. In many cases, a trusted third party is required that acts as central pseudonymization service provider and thus controls access to research data. Furthermore, the approaches are limited to database records only

(k-anonymity), i.e. structured data. Images in the Digital Imaging and Communications in Medicine (DICOM) [77] format were investigated, but the wide-spread Health Level 7 Clinical Document Architecture (HL7 CDA) [57] is often neglected. The focus of this thesis is therefore the investigation of pseudonymization as a holistic security architecture supporting the medical document standard HL7 CDA. In particular, the thesis aims to answer the following research questions:

- Can pseudonymization be used as a holistic security architecture controlling authentication, authorization, and data access to sensitive data?
- How does pseudonymization fare in a real-world scenario with unstructured source data? What are the preconditions for successful pseudonymization?

The main contribution of this thesis is twofold: First, a pseudonymization approach is developed that acts as security architecture for sensitive health data, but should also be applicable to any sensitive data record. We propose to apply cryptographic means to create and protect pseudonyms and rely on pseudonyms as access identifiers, thus creating an access control mechanism based on pseudonym knowledge. This pseudonymization approach therefore has to implement secure storage and retrieval techniques.

Second, the pseudonymization approach is extended to be applied to standardized medical data in the form of HL7 CDA documents. HL7 CDA documents are XML-based and specifically designed to be interoperable among different systems to facilitate data exchange and automated processing. Therefore, the documents are perfectly suited for automated large-scale screening for research activities. Usually anonymization techniques are used in this case, but as already stated, anonymization is irreversible by design which is a considerable disadvantage here. In many cases, the corresponding patients need to be contacted (as further information is required) and thus need to be reidentified. The unstructured nature of many (source) documents such as medical discharge papers also poses a difficulty, which requires that the necessary information first needs to be extracted and converted into HL7 CDA documents in a privacy-preserving way.

1.4 Structure of the thesis

The thesis is structured as follows: In Chapter 2, a general overview of privacy and privacy-enhancing technologies is given. Section 2.1 introduces the term information privacy and sets it into context with security. Section 2.2 defines privacy-enhancing technologies (PETs) and 2.3 defines their properties which are organized into a taxonomy with seven dimensions. Section 2.4 introduces pseudonymization as a privacy-enhancing technology, identifies its properties, and lists related work. In Chapter 3, PERiMETER, a holistic Pseudonymization and pERsonal METadata EncRyption methodology for the concurrent primary and secondary use of health data, is presented. Section 3.1 identifies the requirements with respect to existing pseudonymization approaches and discusses the PET properties of PERiMETER. Section 3.2 describes PERiMETER's core concepts, query mechanisms, and extensions to the core concepts like data integrity

verification. Section 3.3 then describes different application scenarios, followed by a discussion of implementation details and an analysis of the methodology's privacy assurance in Section 3.4. In Chapter 4, MEDSEC, a system for the creation of MEDical records for SECondary use, is presented. Section 4.1 describes the overall requirements and lists related work. Section 4.2 describes MEDSEC's core phases and modules. Section 4.3 provides implementation details and evaluation results of the methodology's accuracy. And Chapter 5 completes this thesis with a summary of the results (Section 5.1), revisited research questions (Section 5.2), a summary of the benefits (Section 5.3), and a discussion of limitations and future research directions (Section 5.4).

This thesis is based on the following peer-reviewed publications:

- Johannes Heurix, Peter Zimmermann[1], Thomas Neubauer, Stefan Fenz: A taxonomy for privacy enhancing technologies, Computers & Security, Vol. 53, pp. 1-17, 2015

- Thomas Neubauer, Johannes Heurix: A methodology for the pseudonymization of medical data, International Journal of Medical Informatics, Vol. 80, No. 3, pp. 190-204, 2011

- Johannes Heurix, Michael Karlinger, Michael Schrefl, Thomas Neubauer: A hybrid approach integrating encryption and pseudonymization for protecting electronic health records, Proceedings of the 8th IASTED International Conference on Biomedical Engineering, pp. 117-124, 2011

- Johannes Heurix, Michael Karlinger, Thomas Neubauer: Pseudonymization with metadata encryption for privacy-preserving searchable documents, Proceedings of the 45th Hawaii International Conference on System Science (HICSS), pp. 3011-3020, 2012

- (invited follow-up journal publication) Johannes Heurix, Michael Karlinger, Thomas Nebauer: PERiMETER - Pseudonymization and pERsonal METadata EncRyption for privacy-preserving searchable documents, Health Systems, Vol. 1, No. 1, pp. 46-57, 2012

- Johannes Heurix, Antonio Rella, Stefan Fenz, Thomas Neubauer: Automated Transformation of Semi-Structured Text Elements, Proceedings of the 2012 Americas Conference on Information Systems (AMCIS), pp. 1-11, 2012

- (invited follow-up journal publication) Johannes Heurix, Antonio Rella, Stefan Fenz, Thomas Neubauer: A rule-based transformation system for converting semi-structured medical documents, Health and Technologies, Vol. 3, No. 1, pp 51-63, 2013

[1]Based on concepts introduced in this paper, Peter Zimmermann has written his master's thesis with the title 'A Survey and Taxonomy on Privacy Enhancing Technologies' which was co-advised by the author of this dissertation.

- Stefan Fenz, Johannes Heurix, Thomas Neubauer, Antonio Rella: De-identification of unstructured paper-based health records for privacy-preserving secondary use, Journal of Medical Engineering & Technology, Vol. 38, No. 5, pp. 260-268, 2014
- Johannes Heurix, Stefan Fenz, Antonio Rella, Thomas Neubauer: Recognition and pseudonymisation of medical records for secondary use, Medical & Biomedical Engineering & Computing, Vol. 54, No. 2, pp. 371-383, 2016

CHAPTER 2

Pseudonymization as a privacy-enhancing technology

In this chapter, an overall introduction to (information) privacy and privacy-enhancing technologies is given and several privacy-related terms are introduced. Then, pseudonymization is characterized as privacy-enhancing technology and related work is described. The content of this chapter has been published in the following publication:

- Johannes Heurix, Peter Zimmermann[1], Thomas Neubauer, Stefan Fenz: A taxonomy for privacy enhancing technologies, Computers & Security, Vol. 53, pp. 1-17, 2015

2.1 Privacy and security

Privacy is usually something which is deemed as highly important to everybody, yet it is surprisingly difficult to define. Historically, the famous phrase of the right "to be let alone" dates back to 1834, originated in the ruling on copyright in the case Wheaton v. Peters by the U.S. Supreme Court [117]. Later, in his dissent of the U.S. Supreme Court's decision in the case Olmstead v. United States in 1928, associate justice Louis Brandeis argued that "the right to be let alone" is "the most comprehensive of rights, and the most valued by civilized men" [118]. Privacy has also been recognized as a fundamental human right in article 12 of the United Nations Universal Declaration of Human Rights of 1948 [112] as well as in article 8 of the European Convention on Human Rights of 1987 [34]. Furthermore, numerous international and national legal acts were introduced to enforce privacy, such as the Privacy Act of 1974 in the United States [113], the European

[1]Based on concepts introduced in this paper, Peter Zimmermann has written his master's thesis with the title 'A Survey and Taxonomy on Privacy Enhancing Technologies' which was co-advised by the author of this dissertation.

Directive 95/46/EC of 1995 [44], as well as individual national implementations like the Austrian Data Protection Act (DSG2000) of 2000 [90].

(Information) Privacy is closely related to *(information) security*. Security is usually described as the composite of the attributes *confidentiality*, *integrity*, and *availability* [84]: Confidentiality refers to the prevention of particular assets being accessed by unauthorized parties. This also includes the pure knowledge of the assets' existence. Integrity ensures that the assets are only modified by authorized parties and that unauthorized tampering can be detected. Availability is the attribute which determines the accessibility of assets to authorized parties at appropriate times. There are other aspects that are related to these security attributes such as *reliability* (correct continuity), *safety* (absence of adverse consequences), and *maintainability* (how easily modifications and repairs can be conducted), forming the concept of *dependability* [8] (Figure 2.1).

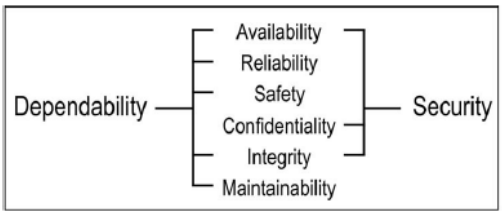


Figure 2.1: Dependability and security [8]

Privacy is often perceived as part of security, in particular as an aspect of confidentiality. Privacy mainly deals with the following three aspects [84]: First, the core of the privacy discussion is always the *affected subject*, i.e. the individual, group, company, or organization that is the target of privacy-related issues. Second, it is not the affected subject itself but *sensitive data* and information about this subject which is privacy-critical. The knowledge of the existence of an individual in general or a company is usually not privacy-compromising, as is publicly available and non-critical information such as a company name or a company's business area. However, sensitive information combined with the data subject is privacy-compromising. For example, the knowledge of the existence of a particular medical condition on its own is considered publicly available information, but the knowledge that a particular individual suffers from this medical condition is usually regarded as privacy-critical. The same is true for incidents such as security breaches: While it is common knowledge that these incidents happen, companies will surely be very cautious and reluctant when disclosing to the general public that they suffered from a security breach. This leads to the third aspect of privacy, namely the *controlled disclosure* of sensitive data, i.e. to control who is allowed (trusted) to have knowledge of critical information about the affected subject.

In his taxonomy, Solove [107] investigates privacy from a legal point of view by identifying privacy problems and defines four principal groups of activities which may create privacy policies. As shown in Figure 2.2, the center of this model is again the data

subject which is the individual that is affected by privacy-compromising activities. Other entities such as other people, businesses, and the government collect information about the data subject, process the information, and disseminate the information. Furthermore, these other entities may directly 'invade' the data subject's life. This results in the following groups of potentially privacy-compromising activities: (i) information collection, (ii) information processing, (iii) information dissemination, and (iv) invasion.

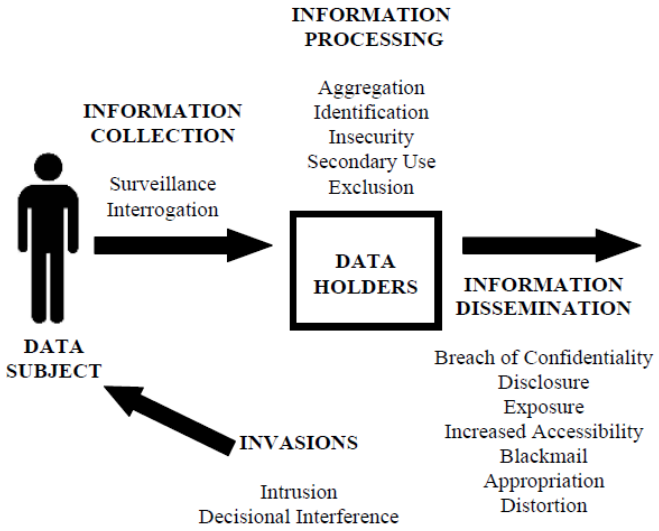


Figure 2.2: Groups of activities that affect privacy [107]

Information collection refers to activities with the goal of gathering information about a particular data subject. This includes passive activities like surveillance where a data subject is either watched, listened to, or being recorded, as well as active activities like interrogations where the data subject is more directly approached in the form of questioning. Information processing involves activities that combine several data pieces about a data subject (aggregation), link information to a data subject (identification), or use information for purposes other than the original purpose for which the information was actually collected without the data subject's consent (secondary use). Information can also be processed with little regard to security-related issues (insecurity) or without informing the data subject about the information collection and processing activities (exclusion). The third group of activities is related to transferring personal information to others. This includes the violation of confidentiality (breach of confidentiality), passing on truthful information (disclosure) or false or misleading information (distortion) about the data subject to others, increasing the possibility to access the information (increased accessibility), and the threat of disclosing personal information (blackmail). In the final group, activities that are directly targeted at the data subject are collected such as invasive activities that interfere with the data subject's tranquility or solitude (intrusion)

as well as the direct invasion into the data subject's decisions regarding private affairs by the government (decisional interference).

Rezgui et al. [91] describe eight dimensions of (Web) privacy which have to be considered in order to preserve privacy: *Information collection* ensures that a data subject's private information is only collected with the subject's explicit consent and not without his/her knowledge. *Information usage* defines for which purposes the private information is used. *Information storage* defines how and for how long private information is stored. *Information disclosure* controls how and to whom private information is disclosed. *Information security* refers to the existence of common security policies and mechanisms that guarantee the security of information. *Access control* states who may have access to which information under which conditions. *Monitoring* ensures that each movement of sensitive information (input/output) is traced and maintained. And finally, *policy changes* could be introduced but must not be done retroactively so that previously collected sensitive information is not subject to the new (less-protective) policies.

2.2 Privacy-enhancing technologies

Privacy can be enforced in multiple ways: On the one hand, privacy can be regulated by law, such as [113], [44], [90]. However, laws can become outdated over time and loopholes and ambiguities may arise, which is especially true for the information privacy sector where technologies and methods of data use change rapidly. Thus, laws need to be amended to match the current situation but not necessarily to the benefit of the individual. Every introduction or modification of existing legal regulations is closely followed by numerous interest groups and lobbies that try to exert their influence to render the outcome more favorable to their clients, usually governmental institutions, commercial enterprises, or other influential organizations. Or legal regulations can more directly be amended or overridden by more pressing concerns, leading to ad hoc legislation, as already mentioned in the previous section (cf. [114]).

On the other hand, privacy can be controlled by organizational measures such as privacy policies commercial companies usually rely on. These policies dictate how to deal with privacy-critical actions and issues such as the disclosure of sensitive personal information about customers. But it has been shown that there are certain issues with privacy policies [106], be it that they are inaccurately defined, contain errors, or simply do not exist. Another issue is that policies were often installed only after a certain (security) incident as a quick response measure, after the damage was already done. As security policies and their use have evolved over the course of time, privacy policies are usually defined and enforced more proactively these days. Even so, organizational methods such as privacy policies suffer from the same shortcoming as regulations by law, namely that these can be modified if they prove to be too restrictive and compromising to business operations. This is especially true for companies whose core business case is that of handling and processing personal data of customers, such as Google and Facebook, companies that have a long history of controversial privacy-related practices. Even if

organizational privacy policies are bulletproof today, one has to trust the corresponding company that these policies will not change in the future to the detriment of customers.

Given these limitations of legal and organizational measures, it is clear that they have to be supplemented with more robust, i.e. technical means. If their foundations (e.g., cryptographic algorithms and their implementations) are proved to be sound, technical measures are dependable tools that ensure that information is collected and processed in a privacy-preserving way. Due to the close relationship of privacy and security, privacy measures are in general already integrated with generic information security-enforcing technical solutions, or privacy preservation is the positive 'side effect' of security measures. So-called privacy-enhancing technologies or PETs (cf. [48]) are technical means specifically targeted at preserving and enhancing privacy. With privacy-enhancing technologies, security aspects like confidentiality, integrity, and availability are still applicable, but extended with attributes that reflect the focus on privacy. In a collaborative work [83], some key privacy-specific aspects are described as follows: From an attacker's perspective, *anonymity* (Figure 2.3) is defined as the property of an individual not to be sufficiently identifiable by an attacker within a particular set of individuals, the anonymity set. The opposite is denoted as *identifiability*. Here the authors emphasize the word 'sufficiently' which allows anonymity to be quantifiable (cf. [99]). Furthermore, it is obvious that an individual (or subject) can only be anonymous within a set of individuals that have the same role. For example, considering senders and receivers of messages, an anonymous sender can only be anonymous within a set of senders, but not within receivers. Since this anonymity set may underlie various changes over time (as anonymity depends on the context of the individual), the authors also define the term *anonymity delta* which measures the difference in an individual's anonymity due to various observations by an attacker that change the attacker's knowledge of the context. As anonymity can never increase, the delta is always negative.

Another concept defined in [83] is *Unlinkability* which further specifies the relationship of entities (e.g., individuals, messages) within a defined system that cannot be distinguished by an attacker, or in other words, an attacker cannot identify whether these entities are related to each other. In contrast to this, *linkability* refers to the fact that an attacker is able to sufficiently distinguish if the entities are related to each other. Again, a delta measures the changes of these properties due to observations of the attacker. Anonymity can be described in terms of unlinkability if the relationships between entities of different types are taken into consideration. Going back to the example of senders and receivers, sender anonymity is achieved when different messages of the particular sender cannot be linked to the sender. A more fundamental property is *undetectability* which states that an attacker cannot sufficiently decide on whether a particular entity actually exists or not. *Unobservability* (Figure 2.4) combines anonymity and undetectability such that an entity is unobservable when it is undetectable against all subjects not involved in it and when the subject(s) which is (are) involved in the entity is (are) anonymous against other subject(s) involved in that entity.

Finally, *pseudonymity* refers to the usage of *pseudonyms* as identifiers where a pseudonym is an identifier of a subject which is not the subject's real name. Pseudonymity

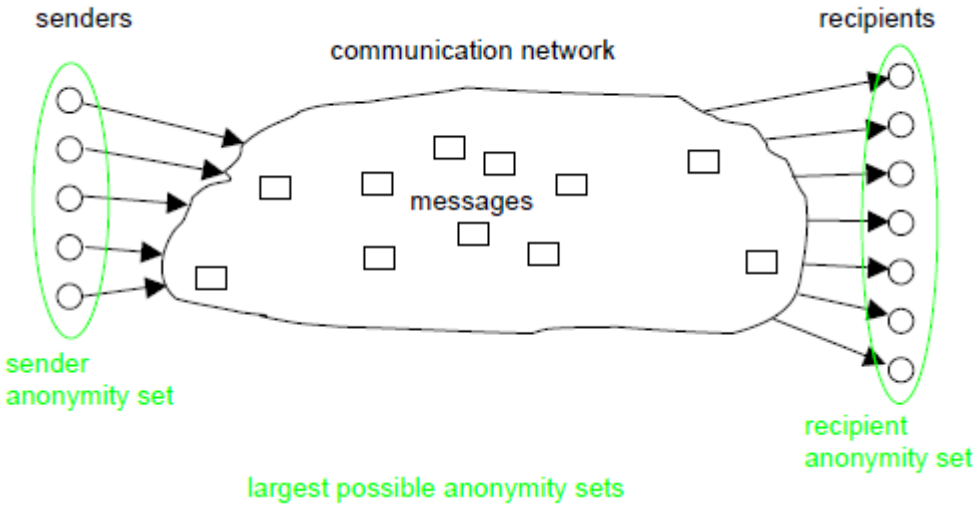


Figure 2.3: Anonymity sets within the a sender/recipient-sets [83]

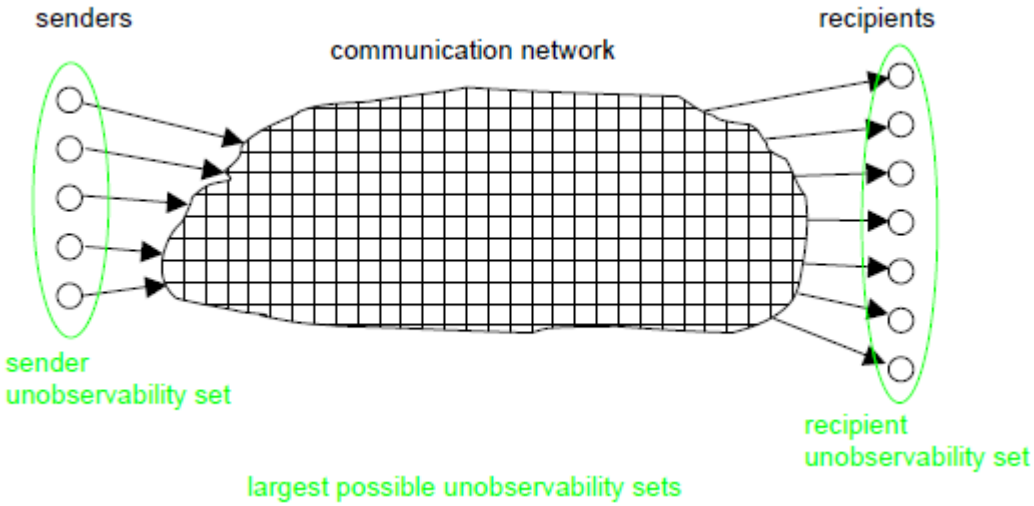


Figure 2.4: Unobservability sets [83]

also incorporates aspects concerning the creation and use of pseudonyms, also including the controlled disclosure of a pseudonym holder's real identity by so-called identity brokers. Pseudonyms can be assigned to individuals or can be related to groups of individuals. Pseudonyms might also be transferable, depending on the application scenario (a more detailed analysis of the pseudonymity property can be found in the following section).

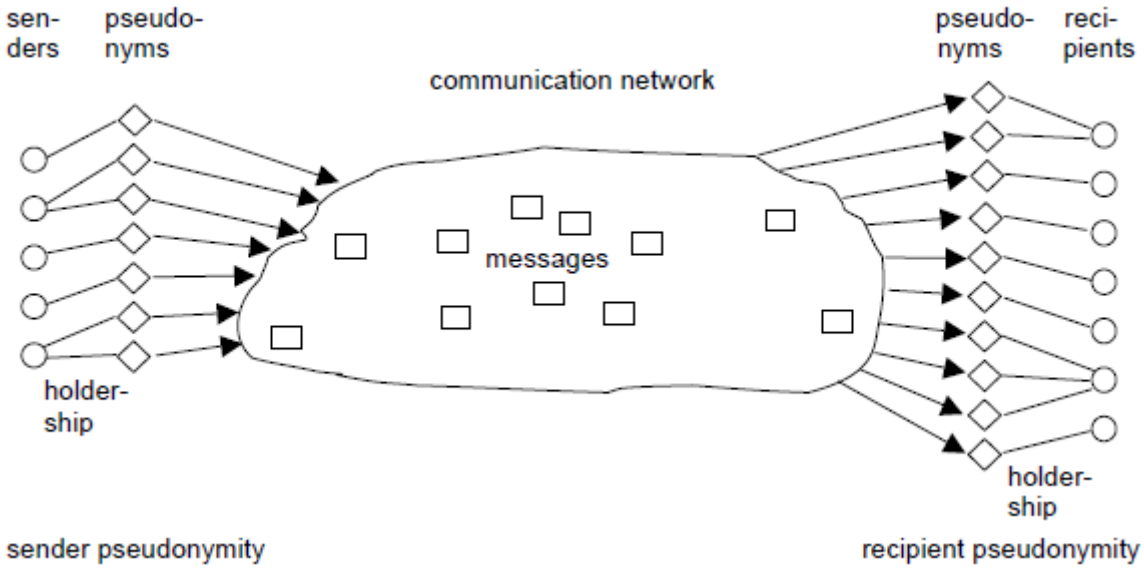


Figure 2.5: Pseudonymity sets [83]

In the following section, common properties of privacy-enhancing technologies are discussed.

2.3 Properties of privacy-enhancing technologies

In the last decade, numerous PETs were proposed that are targeted at different aspects of privacy including network traffic anonymization (e.g., TOR [40]), identity management (e.g., IDEMIX [24]), or anonymous data storage (e.g., Free Haven [39]), just to name a few, all with the ultimate goal of protecting the individual's privacy but based on different building blocks such as cryptographic primitives or separation of information. Since privacy is a many-faceted concept, reflected by the different application areas of PETs, categorizing PETs is a non-trivial task. Still, there are several common properties of PETs which can be used to classify them. We have identified a set of PET properties and collected them under seven dimensions (Figure 2.6), ultimately creating a taxonomy:

Scenario The *Scenario* dimension defines the primary untrusted actor and potential attacker in a privacy-sensitive information exchange operation.

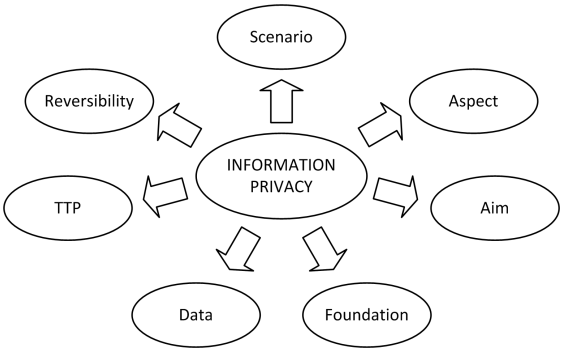


Figure 2.6: Dimensions of information privacy-enhancing technologies

Aspect The *Aspect* dimension defines which privacy aspect is addressed by the PET.

Aim The *Aim* dimension defines the PET's aim or means of privacy.

Foundation The *Foundation* dimension defines the underlying security model and strength of the PET.

Data The *Data* dimension defines the type of data that is addressed by the PET.

Trusted Third Party The *Trusted Third Party* dimension defines the necessity of a trusted third party and its involvement in a PET.

Reversibility And finally, the *Reversibility* dimension defines if and under which circumstances a PET operation is reversible.

For the remainder of this chapter, we use the notions of service or data *consumers* and service or data *providers* to distinguish actors in a client-server interaction scenario: A service consumer is a client who requests a particular service from a server acting as the corresponding service provider. This service may cover operations such as the execution of query operations of outsourced databases or the provision of authentication services. In this context, especially when data-at-rest is considered, there must be a clear distinction between the *data owner*, *data originator*, and *service consumer*. To distinguish between data owner and data originator, we define the data owner as the entity responsible for storing and processing a piece of data (e.g., database owner), whereas the data originator is the data subject (e.g., personal information about a person stored in a database). The service consumer in turn is an entity requesting data about the data originator. This definition indicates that the data originator does not necessarily need to be the data owner or service consumer, depending on the data usage scenario.

2.3.1 Scenario

The first dimension *Scenario* (Figure 2.7) describes the primary actors of a protocol or privacy-critical scenario and categorizes them from the trust perspective (cf. [70]).

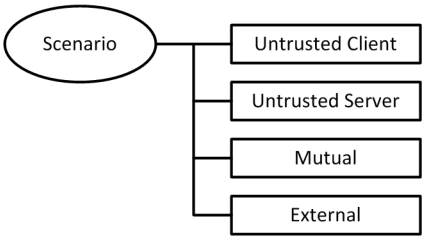


Figure 2.7: Scenario dimension

In the *Untrusted Client* scenario, the service consumer is regarded as untrusted and as a potential attacker. Since the main goal of attackers of privacy-relevant services is to acquire more information from sensitive elements than intended, we apply the honest-but-curious model as our primary attacker model (cf. [52]), a model which is used especially in the analysis of secure multi-party computation. In this model, the adversary follows the protocol faithfully but tries to find out more information than intended by analyzing the information flows. Analogously, the untrusted instance is actually semi-trusted (though we stick to the untrusted notion for clarity reasons). A typical attacker of this type is a user who tries to find out the identities of a particular person in an anonymized database. The classic k-anonymity [99] approach and two well-known extensions l-diversity [69] and t-closeness [66] deal with this kind of attacker. K-anonymity aims to provide indistinguishability of individuals whose data is stored in a database by grouping them to equivalence lists with common quasi-identifiers through generalization and suppression techniques as follows: At first, the quasi-identifiers are determined which are person-identifying properties that are not identifying on their own but in combination. For example, the postcode of an address is not identifying on its own, but it is in combination with the street name and house number. Then the quasi-identifiers of each person are generalized so that multiple persons share the same quasi-identifiers (e.g., removing the least-significant numbers of a postcode). If quasi-identifiers are too specific so that applying generalization would result in a disproportionate loss of data accuracy, the elements can also simply be removed (suppression). The result is a set of multiple groups whose members share the same quasi-identifiers but have different so-called sensitive attributes (e.g., illness). The extensions of the basic k-anonymity approach specifically consider background knowledge and the distribution of sensitive attributes to prevent certain statistical attacks. The overall goal is to grant untrusted users access to the anonymized databases (for, e.g., research purposes) without disclosing the identities of the data originators. Without further background knowledge than provided in the k-anonymized database, an attacker is not able to unambiguously identify a certain person within the database, apart from the fact that the person is present in this database and part of a certain equivalence group. Due to the equivalence group containing multiple manifestations of the sensitive attribute, it cannot be unambiguously assigned to a particular person.

A malicious adversary by contrast is expected to act arbitrarily. In certain cases,

the honest-but-curious model is not sufficient for ensuring privacy, especially when authentication is required. In this case, PETs need to be supplemented with additional security measures (e.g., public-key authentication). In both cases, the focus of an attack obviously lies on the content of the service the server provides, not the server's identity.

The *Untrusted Server* scenario describes the case of a service provider that aims to gain more information about the service consumers than necessary. This usually relates to the identity of the service consumers, leading to techniques such as anonymous communication which can be realized by relying on intermediate proxies between a sender and a receiver responsible for masking the sender's identity (cf. [28]). The main issue with anonymous communication is to ensure or verify the rights of a particular service consumer. This problem can be solved by using some form of anonymous authentication in the form of pseudonyms (cf. [48]) or other forms of anonymous credentials (cf. [22]) that enable the authentication of a particular service consumer (i.e. verify his/her service consumption rights) without disclosing his/her actual identity. Apart from the service consumer's identity, a server could also be interested in the data the service consumer is requesting. In the case of a data service provider, the straightforward solution for this issue is client-side encryption before uploading records to the server. The difficulty here is to support efficient querying over the encrypted data, which can be solved by applying specific cryptographic algorithms (e.g., [16]). Another potentially sensitive element is the access pattern of data of service consumers.

Private information retrieval (PIR) [31] protects the privacy of the querying user from the database server so that the server cannot identify the piece of data that the user is interested in. Apart from the trivial solution of copying the whole database content as the result set of a query to the client, two general approaches exist with different security models: The first approach requires multiple servers ($n \geq 2$) where the data is replicated and which are queried and then return the query results. The combined results are then processed at the client without the server having knowledge of the actual piece of data the data consumer is interested in. The underlying algorithms can rely on techniques such as secret sharing [102] to make sure that servers do not gain unintended information. In this form, PIR provides perfect or information-theoretical security, as long as the databases are non-colluding. When only a single server is involved, only computational security is possible. Single PIR solutions rely on hard mathematical problems [65] to hide the desired data from the server.

The *Mutual* scenario refers to a scenario where both actors cannot trust each other. This is usually the case when there is no clear distinction between service provider and service consumer or when a set of equal actors intends to achieve a common goal without revealing too much to each other. *Mutual* is also used for typical communication scenarios of equal partners (Alice and Bob), who do not fully trust each other. For example, in oblivious transfer [87] a secret is transmitted between a sender and a receiver without the sender knowing what the receiver has actually received. In the original proposal, the receiver gets the correct message with a probability of 1/2. This has been generalized to 1-out-of-2 oblivious transfer where the sender sends two messages and the receiver gets only one, but the sender does not know which one and the receiver has no knowledge

of the other message's content [45], and later to 1-out-of-n oblivious transfer with n possible messages [18]. So the goal of oblivious transfer is to provide indistinguishability of the selected message, while the actual message content is kept unmodified. Oblivious transfer protocols share certain properties with single database PIR, such as the reliance on number-theoretic problems and the property of hiding the behavior of the receiver (selected piece of data). In fact, it has been shown that there is a direct relation between single database PIR and oblivious transfer [35].

The last scenario *External* refers to an external agent as primary threat to the primary actors' privacy. This usually is the case for communication protocols where the communication partners use PETs to protect themselves from an external adversary such as an eavesdropper intercepting encrypted messages (Alice, Bob, and Eve) or an adversary coercing the communication participants to decrypt certain messages. The aim of deniable encryption is the ability to deny the knowledge of the secret key and thus the plaintext data [26]. In the original proposal, the focus lies on public key-based sender-deniable encryption schemes, where receiver-deniable and sender-and-receiver-deniable schemes can be constructed from the purely sender-deniable scheme by adding XOR operations. The foundation of the scheme revolves around the selection of an element (from a predefined domain) by the sender either truly randomly or from a pseudo-random distribution, where the receiver is able to distinguish between these two cases, while an attacker cannot. Shared-key deniable encryption can be realized by using an OTP or creating fake messages in advance. Receiver-deniability requires interaction, while for sender-and-receiver deniability, the protocol requires the involvement of multiple intermediaries where at least one of them must remain uncoerced. In [81], a bi-deniable protocol that allows both sender and receiver to be simultaneously coercible without the need for trusted third parties was proposed.

2.3.2 Aspect

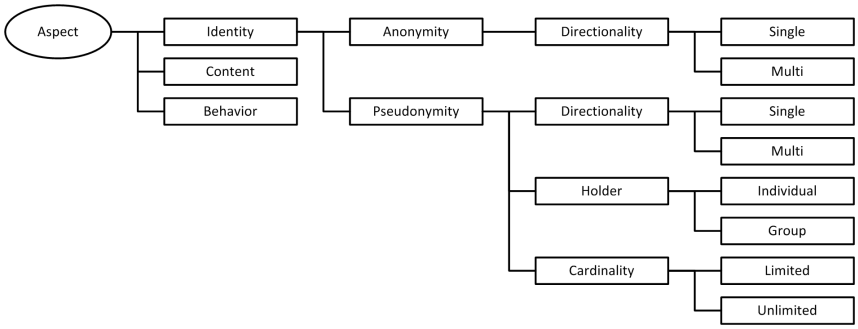


Figure 2.8: Aspect dimension

The *Aspect* (Figure 2.8) dimension covers the general target of the PET (cf. [48]) and distinguishes between *Identity*, *Content*, and *Behavior*. Here, the privacy aspects

that were already mentioned in the previous section are categorized in more detail and logically supplement the *Scenario* dimension.

Identity is the primary aspect of privacy and refers to hiding or masking the identity of involved persons. This includes 'active' actors such as service consumers requesting a certain service from a service provider while hiding/masking their identities as well as 'passive' users whose data is stored or processed by the service provider and requested by third parties as data consumers. While in general the first type of identity protection can be enforced more easily (ultimately by choosing not to consume a particular service), the latter requires that (i) either the data stored about the data originators, which is made available to service consumers, is modified in a privacy-preserving way or (ii) this data is already stored following the principles of data minimization. Identity protection can be enforced by either *Anonymity* or *Pseudonymity*.

Anonymity refers to being not identifiable within a particular set of entities. In k-anonymity [99], an individual cannot be unambiguously identified within the corresponding equivalence group. If an attacker concludes that a particular individual is part of an equivalence group by matching the generalized quasi-identifiers, the attacker can identify the correct entry for the sensitive attribute only with a probability of i/k, where i is the number of the different entries for the sensitive attribute within this equivalence group and k the number of members within the group (with $i < k$ and usually $k > 1$). In general, anonymity within stored data requires the diligent identification and removal of person-identifying elements such as names and addresses. Here, the problem is to find out which elements actually are identifying. Names are no-brainers, but what about the age? There is no general answer to this question since it depends strongly on the context, i.e. how many individuals are present in a data set, what kind of information is stored about them, and the domain the data records are used in. For example, the HIPAA Privacy Rule [116] defines the Safe Harbor de-identification method for creating data sets open for disclosure which dictates to remove a fixed set of 18 so-called Protected Health Information (PHI) including names, phone numbers, and email addresses. Although this seems to effectively counter the risk of reidentification, there are still some issues with this approach. On the one hand, removing elements from a data set will obviously reduce the data set's expressiveness due to data loss. On the other hand, removing all these PHI elements does not necessarily guarantee true anonymity within the scrubbed data set [10]. Background knowledge in the form of publicly available registers (e.g., voter registries) or the distribution of data variables (e.g., the number of people within the set with a particular medical condition) allows statistical or inference attacks and thus increases the risk of successful reidentification. HIPAA also defines a second method for de-identification which requires a domain expert to decide on the critical PHI (Expert Determination) and consider other context-specific requirements. By combining statistical or scientific principles for de-identification with domain knowledge, the expert can calculate the acceptable risk level of reidentification while maintaining the data usefulness by balancing privacy and data expressiveness. Apart from completely removing identifying elements, their values can also be generalized to more general levels. In that case, individuals have the same PHI combinations and are thus hidden within the groups

with a common set of PHI values (cf. equivalence sets [99]). Again, this increases overall data expressiveness but is also prone to statistical and background knowledge attacks.

Pseudonymity is defined as relying on pseudonyms as identifiers [83]. Pseudonyms are often used in medical research where the identities of the involved patients need to be kept confidential towards the researchers (e.g., [79], [51], [60]). Before researchers are granted access to the medical data, the identifiers of the patients are replaced with generated pseudonyms so that intended traceability and linkability can be controlled (e.g., identifying all medical records of a particular patient), depending on the requirements. Another concept that hides a person's identity is an anonymous credential system, but in the different context of user authentication by proving the possession of a certain credential without disclosing the user's identity, introduced in [29]: A sender uses different pseudonyms for the communication with a credential provider and a credential receiver, where the pseudonym for the receiver is blinded before it is sent to the credential provider. This basic concept was extended to a practical protocol in [22] which also discourages sharing of pseudonyms by different users (non-transferability), and supports single or multiple uses of the credentials and optional reversibility: Non-transferability by sharing the credential owner's master secret with someone else is discouraged since it allows the other user to use all other credentials of the owner (all-or-nothing non-transferability). Alternatively, a certificate authority (CA) is responsible for checking the owner's external public key registered at the CA (PKI-assured non-transferability). While the certificates are multi-use in nature, the protocol can be modified so that multi-use of a credential can be detected by the verifier without the user showing the actual credential, which is a cornerstone of applications such as online e-cash.

Both *Anonymity* and *Pseudonymity* have the property of *Directionality* which indicates whether anonymization/pseudonymization is single-sided or multi-sided. For example, k-anonymity [99] is a typical single-sided anonymization protocol; the server as data provider is obviously known to the clients. In contrast, in an anonymized network (e.g., mix nets [28], see below), both the senders and receivers can be anonymized, which results in a multi-sided anonymization scheme.

While identifying properties from individuals are removed (or generalized) in an anonymous data set, identifying properties are replaced with an identifier that does not allow unintended relinking to the corresponding individual in a pseudonymous set. This allows more finely controlled unlinkability between data records by selection of the proper count and reusability of pseudonyms. This is considered by the *Pseudonymity* attributes of *Holder* and *Cardinality*: *Holder* indicates the number of individuals that are referenced to a single pseudonym and can be either *Individual* for a 1:1 relationship between pseudonym and individual or *Group* for group pseudonyms shared between all members of a particular group, indicating a 1:n relationship. By carefully selecting the number of pseudonym holders, the 'level of anonymity' can be controlled across different groups. The *Cardinality* attribute expresses how many pseudonyms can be used by a single individual, either *Limited* (usually a single pseudonym per individual) or *Unlimited*. The *Cardinality* property allows to control the traceability of individuals or transactions. The usefulness of pseudonyms is reflected, e.g., by the application of pseudonymization

in areas such as secondary use of medical records for research purposes where the identities of patients are kept hidden while records of the same patient should still be identifiable as such (e.g., [86], [111], [79], [60]). Anonymous credentials as mentioned above are single-sided (credentials protect the credential owners, not the verifiers) and each pseudonym is used by only a single credential holder. Depending on the usage scenario, each credential holder can use one or multiple pseudonyms (if it is necessary to unlink multiple authentication operations).

Content refers to hiding or masking the data content of a service consumed by a client and corresponds to the classic security attribute of confidentiality. While privacy usually refers to protecting the identity of the persons involved, *Content* refers to the data processed or created during the service consumption, including both payload and meta data. The main approach to hiding sensitive content in both data-in-motion and data-at-rest scenarios is to employ encryption. The aim of encryption of data-in-motion is to prevent unauthorized third parties from gathering sensitive details of the content exchanged between communication partners and does not require further processing other than encryption and decryption. In the case of encrypted data-at-rest, querying and processing methods are desirable that are more efficient than the trivial solution of retrieving the encrypted data, decrypting it, and then performing the operation over the decrypted data, which makes a simple querying operation a more complex task. Searchable encryption techniques allow the server to execute limited operations over encrypted data without the need for decryption by the server. A potential solution is to create special search index structures by individually encrypting or hashing keywords or attributes for exact match queries [37], where the latter counters inference attacks by carefully choosing the cardinality of the algorithm's co-domain. This allows to query for predetermined attributes within a data table only. A public key-based searchable encryption scheme was first presented in [16] which allows an untrusted server to verify the existence of an encrypted keyword that has been encrypted with a receiver's public key by a sender. The server is provided with a trapdoor for this particular keyword that is created with the receiver's secret private key. Obviously, the payload data encryption can be fully reversed, but only by the user with access to the secret key (cooperation of the key owner required).

The third element of *Aspect* is *Behavior* which refers to hiding the behavior of actors. For example, in contrast to *Content*, not the content of data records retrieved by a service consumer is relevant (e.g., because data is encrypted) but which data records were retrieved. Again, techniques that are more efficient than the trivial solution of retrieving the whole database are considered. A private information retrieval scheme [31] effectively hides the access pattern of a data consumer insofar as the query operations executed by the data provider do not yield the actual results but only intermediate results. The final result is then calculated and determined at the data consumer's side. In general, not only the payload data itself but also any meta data needs to be considered for successfully hiding the data consumers' behavior.

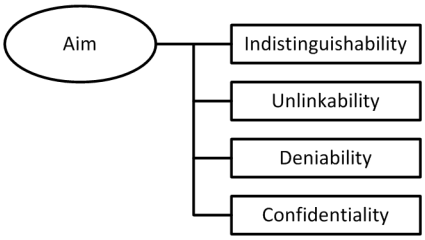


Figure 2.9: Aim dimension

2.3.3 Aim

The *Aim* (Figure 2.9) dimension describes how privacy is achieved or which aim a particular PET is pursuing. The properties cover general terms that are usually associated with a PET (cf. [83]).

Indistinguishability is the property of an entity that makes it impossible to unambiguously distinguish it from another entity in certain regards. This refers to hiding an individual within a specified set of individuals and, e.g., modifying the attributes in a data table to make them indistinguishable from each other. In k-anonymity [99], individuals within the same equivalence group share the same generalized (or suppressed) quasi-identifiers so that the individuals cannot be distinguished from each other. This is only true for individuals within an equivalence group; members of different equivalence groups have different quasi-identifiers and can therefore be distinguished. The data owner needs to keep a balance between indistinguishability (and thus the data originators' privacy) and data expressiveness for the service consumers by carefully selecting the generalization and suppression strategies.

Unlinkability is closely related to *Indistinguishability* and refers to the entities' property of being not linkable/associable with other entities such as transactions or data entries. In other words, *Indistinguishability* denotes that an entity cannot be distinguished from another entity of the same entity class (e.g., persons in a database), while *Unlinkability* indicates that an entity cannot be related to another entity where the entities need not necessarily be of the same class (e.g., person and corresponding medical data). An example of unlinkability can be observed in anonymous networks such as in a mix net [28]. A mix net is a concept for anonymized communication between sender and receiver and is based on the encapsulation of a message with multiple encryption layers to hide the message's route, combined with mixing the messages (sequence, delay, dummy traffic). Each node within the net is able to decrypt one layer of the message to identify the next node to which the message has to be sent, where multiple messages are collected at each node and sent in different sequences, delayed and extended with dummy messages to break up the correlation of the message path. Each node only knows the immediate predecessor and successor nodes, which effectively unlinks the messages from the sender (and receiver in case of anonymous return addresses) under the assumption of non-colluding servers. Widely-used applications based on the original mix net concept - like the onion routing application Tor [40] - modify certain aspects such as the inclusion of

symmetric cryptography or the determination of the path.

Another example of unlinkability can be found in group signatures [29]: In a group signature scheme, members of the group are able to create a valid signature while the actual individual is kept hidden from the verifier, i.e. the aim is to create signatures which are unlinked from the actual signer and only linked to the group, as well as signatures unlinked from each other when created by the same individual. The initial schemes require the involvement of a trusted group manager of different degrees depending on the scheme for setup, operation, and reversal (i.e. opening a signature) in case of a dispute, and all suffer from a linear increase of the group's public key size depending on the number of group members. A more practical version with a fixed-size public key was proposed in [23] where a group manager is responsible for setting up the group including the creation of the public verification key and member secret keys, as well as for the opening of individual signatures for identifying the corresponding members directly, but who is not able to fake a valid signature. Further works introduce efficiency improvements such as shortening the signatures [15].

Deniability refers to the property of being able to plausibly deny a fact, possession, or transaction, which is the direct opposite of accountability. For example, deniable encryption allows to hide the real cleartext message by providing convincing fake cleartext messages and keys when an adversary has access to the ciphertext message [26]. Thus the attacker cannot unambiguously identify the correct message and key from the fake messages and fake keys. Another potential application is that of hiding personal search queries by slightly modifying them and adding noise in the form of cover queries (e.g., [73]).

The final attribute, *Confidentiality*, corresponds to the secrecy of a data fragment's content. In contrast to *Indistinguishability* and *Unlinkability* where data fragments must not be associable to ensure privacy while being uncritical on their own, *Confidentiality* refers to the requirement of keeping a data fragment protected from unintended disclosure. This is a stricter requirement compared to *Indistinguishability* and *Unlinkability*. *Confidentiality* is usually achieved by some form of encryption. As the decryption process requires the secret key, a key distribution mechanism and/or key management system is required. (Plain) Public key cryptography solves the problem of the necessity to share a secret key, but is relatively inflexible because the data needs to be re-encrypted with a new public key for each new data recipient by the data originator. Proxy re-encryption is a cryptographic protocol proposed in [13] which allows the re-encryption of data by a third party so that the data initially encrypted with the sender's public key can then be decrypted with the receiver's private key, without the third party having access to any of the secret keys or the data's content. This is useful for data exchange via a third party without the need for sharing a common secret key. In [5], proxy re-encryption is the basis of a secure file system. In this scenario, data at a storage provider is encrypted with (symmetric) content encryption keys which in turn are encrypted with a master public key (owned by the data originator). Access is controlled by a semi-trusted third party which is able to re-encrypt the content encryption keys for the data recipients without having access to the data originator's master secret key or the plaintext content

encryption keys. Therefore, trust requirements on the third party concerning the data content can be relaxed. This re-encryption process can only be done correctly for the recipients who are chosen by the data originator in the form of delegation keys which were created using the master secret encryption key and which were then forwarded to the third party.

2.3.4 Foundation

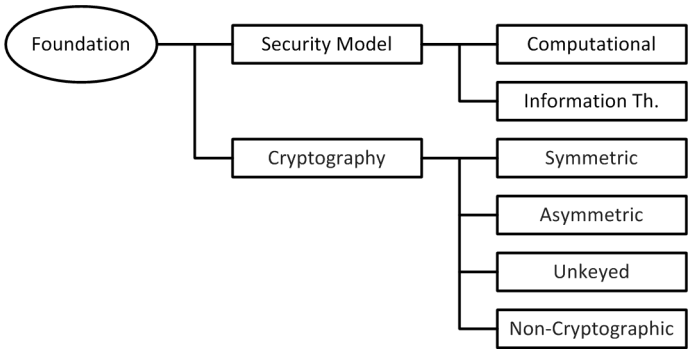


Figure 2.10: Foundation dimension

The *Foundation* (Figure 2.10) dimension describes the conceptual and technical foundation of the PET derived from the general security domain (cf. [120]) and has two attributes, *Security Model* and *Cryptography*.

The *Security Model* attribute is divided into the *Information-Theoretic* and *Computational* models. The *Information-Theoretic* model is based on the information theory introduced in Shannon's seminal work [103] where he proved that encryption with a one-time pad provides perfect security. An information-theoretically secure technique is said to be secure (i.e. unbreakable) even with unrestricted computational resources, or in other words, does not provide any more information even with access to unlimited computational power. For the one-time pad example, this is only true when the key length equals the length of the message and the key is selected in a truly random fashion. Information-theoretic or unconditional security is a desired property in cryptography [71] to keep content confidential, but there are also other areas where the information-theoretic security model is applicable. For example, when a user intends to hide from the data provider which particular pieces of information stored in a database he/she is really interested in, the trivial solution is to transmit the whole database content to this user, exchanging security with increased communication costs, though more sophisticated solutions have been proposed. Multi-server PIR solutions [31] provide information-theoretic security if the servers are non-colluding.

Still, as demonstrated by the one-time pad, information-theoretic solutions are in general less practical, although one-time pads have been used in highly critical message exchange scenarios (e.g., government-level communication). However, the need for more

practical solutions has led to weaker but nevertheless 'reasonably' secure solutions under the *Computational* model. A computationally (or practically) secure solution cannot be broken by a computationally-bound adversary (e.g., polynomial time adversary). Most modern day cryptographic algorithms are based on the *Computational* security model [72] and rely on hard problems, such as number-theoretical problems (e.g., integer factorization), which cannot be efficiently solved without having a certain piece of information (trapdoor), and therefore act as one-way functions. These problems allow shorter key lengths independent of the message length and are thus better applicable than pure information-theoretic solutions. Practically any PET relying on modern day cryptography is based on computational security.

PETs can rely on different types of cryptographic primitives [72]: *Symmetric* or secret-key algorithms are characterized by the shared secret key that has to be known to all communication partners, which requires a secure key exchange and management scheme. *Symmetric* algorithms include block and stream ciphers that apply transformation operations such as substitution and transposition (permutation) of symbols in the plaintext to create the ciphertext. In contrast, *Asymmetric* or public key algorithms rely on a publicly known and a secret part of keys to encrypt and decrypt messages where the secret part acts as the trapdoor to reverse the number-theoretic one-way function (encryption). As each communication party is provided with its own keypair, there is no need for secure key exchange, as long as the public components are authenticated. Apart from that, asymmetric ciphers are known to be much slower than their symmetric counterparts. This issue is usually solved by combining asymmetric and symmetric encryption as follows: The faster symmetric cryptography is applied to the data content that needs to be kept confidential, while the symmetric key is encrypted by asymmetric cryptography to prevent the problem of sharing the secret key. This hybrid encryption technique is widely applied, such as in the SSL/TLS protocol or disk encryption [5]. *Unkeyed* algorithms implement one-way functions that do not have a trapdoor function, and are thus considered irreversible. Unkeyed algorithms, such as hash algorithms, are often combined (or included) in higher level cryptographic algorithms and are designed to be very efficient in terms of computing. Considering this and the fact that the transformation function always yields deterministic outcomes, *Unkeyed* algorithms are not security-enhancing on their own. Again, PETs may rely on individual or a combination of multiple cryptographic primitives to achieve their goal. In the searchable encryption method described in [37], symmetric encryption is used to protect the data content, while individual attributes are hashed to create search indices which can be quickly created and verified due to the hash algorithm's speed. Finally, PETs can also completely forgo cryptography but this only works when certain requirements are met, such as that the original (plain) data is considered not to be available to the data consumers (cf. [99]) or that multiple service providers are non-colluding (cf. [31]).

2.3.5 Data

The *Data* (Figure 2.11) dimension describes what types of data are addressed by the PET or how they are affected. The basic distinction is made according to the general

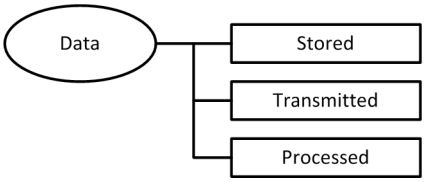


Figure 2.11: Data dimension

states of data, data-at-rest and data-in-motion (cf. [67]).

Stored data refers to data-at-rest in storage areas such as databases, file systems, or other storage methods (e.g., online storage systems). The data fragments are modified to prevent a privacy compromise by an attacker. The adversary can be both service provider and service consumer depending on the data usage scenario, and data modification measures include techniques like client-side encryption of the payload data to protect against untrusted service providers or data anonymization against untrusted service consumers (cf. [99]). Modifications to the data content are preferred over modifications to the access routine since untrusted service providers are expected to have full control over the data they have been provided with anyway, whereas untrusted service consumers should in general have the same access and retrieval means available as fully trusted consumers.

Transmitted data refers to data-in-motion passing one or more stations in a particular protocol, which is a critical component of a PET. These stations do not modify the actual content of the payload data, but are required to process metadata for the successful application of the PET, such as the individual proxies in a mix network that each add and remove cryptographic layers [28]. These stations can be fully trusted (e.g., in the form of trusted third parties), but need not be. Data transferred in an oblivious transfer scheme (cf. [87], [18]) also falls in the *Transmitted* category: While the messages are modified according to the oblivious transfer scheme, the actual message (information) content remains unchanged.

Technically speaking, a service provider storing encrypted data for a service consumer can also be interpreted as an intermediate station since the (cleartext) content would not be modified by the service provider in case of client-side encryption. Therefore, we need to specify transmitted data more precisely as follows: i) data that is transmitted from a sender who is not the same entity as the recipient and ii) data that is transmitted within a reasonable amount of time with the purpose of forwarding a piece of information from the sender to the receiver without the goal of permanent storage.

In contrast to *Transmitted*, a PET addresses *Processed* data when the exchanged payload data content is explicitly processed and modified by the protocol participants. For example, anonymous credentials are data elements whose payload (content) are the authentication tokens produced in the protocol [29]. This example demonstrates the ambiguity of what the payload of a protocol actually is: In the context of anonymous credentials, the authentication token is regarded as payload, whereas in another scenario where the token is used as authentication means for some other service, the token is only

a piece of metadata required to retrieve the actual payload in the form of the database result set. The PET-specific interpretation of data and the corresponding categorization are critical for an accurate analysis of the PET.

2.3.6 Trusted Third Party

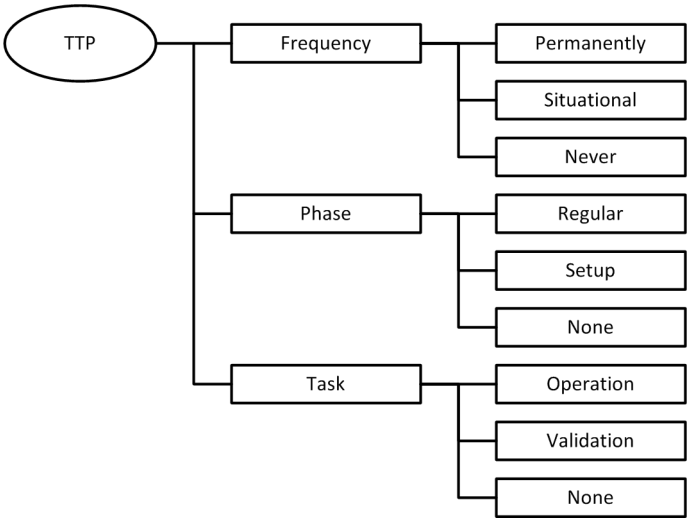


Figure 2.12: Trusted third party dimension

The *Trusted Third Party (TTP)* (Figure 2.12) dimension describes if the PET requires the assistance of a specially trusted entity (cf. [120]) and how this TTP is involved in a PET operation. A TTP's task is to act as a broker or mediator between two or more parties to provide security- and privacy-critical support functions such as user registration and authentication (certification authority). The attributes of the *Trusted Third Party* dimensions are relatively similar to each other, but describe the TTP involvement from different viewpoints.

Frequency denotes how often the TTP interacts with the PET participants: *Permanently* indicates that the TTP's involvement is permanent and that the TTP is an integral component of the PET's operation. In other words, the unavailability of the TTP in this case would prevent the PET operation altogether. By contrast, *Situational* implies that a TTP's involvement is only required in certain situations and scenarios such as in case of a dispute. Apart from that, the PET is not critically affected by a temporary unavailability of the TTP. *None* means that no TTP involvement is required.

Phase describes in which specific phases of PET protocols a TTP is involved: *Regular* indicates that the TTP is involved in the regular operation of the PET, whereas *Setup* refers to the preparation or setup phase required before the actual start of a PET operation. This includes setup phases such as the creation, binding, and distribution of cryptographic keys in privacy-preserving authentication schemes (e.g., [19]). Similar to

Frequency, the PET's TTP phase attribution is *None* if no TTP is involved in any phase of PET execution.

Finally, *Task* determines a potential TTP involvement by describing which task the TTP fulfills: Here *Operation* refers to some regular task that is delegated to the TTP, while *Validation* defines the TTP involvement only in case of validation (or revocation) when the need occurs (e.g., disputes, unsatisfied conditions). Again, *None* represents no TTP at all.

This *Trusted Third Party* dimension helps to clarify ambiguities of certain PET application scenarios. Given a (semi-)trusted third party as intermediate between two untrusted communication parties, the trust statement as defined in *Scenario* can be set to *Mutual* when both communication parties do not trust each other, or *Untrusted Server* when describing the relationship between one of the communication parties and the intermediate server. A particular example for this case is the proxy re-encryption technique where a (honest-but-curious) third party provides re-encryption services for different clients that need not share mutual trust [5]. The *Trusted Third Party* dimension helps to precise the TTP's purpose within this PET scenario. In mix nets [28], the servers within the network serve as (semi-)trusted third parties who are responsible for forwarding the clients' messages. These servers are permanently active and are therefore a core requirement for the operation of a mix net. Other PETs, such as k-anonymity [99] or searchable encryption [37], do not require TTPs at all; the only participants in these PET schemes are the data providers and data consumers. And for some PETs, the involvement of a TTP is dependent on the actual implementation of the PET and can include a permanent involvement of the PET for each operation, a limited or situational involvement such as only for the setup phase of the PET scheme, as well as no involvement whatsoever (e.g., [30]).

2.3.7 Reversibility

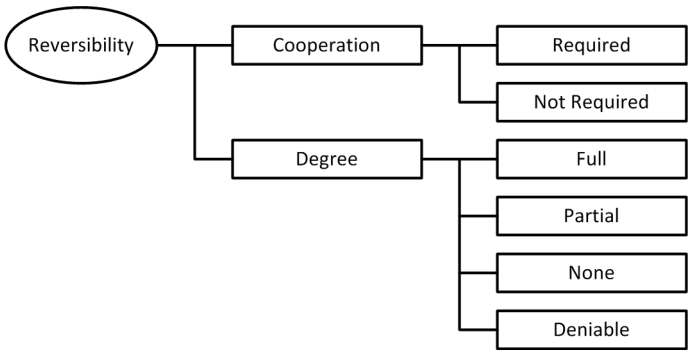


Figure 2.13: Reversibility dimension

The final dimension *Reversibility* (Figure 2.13) describes whether a particular PET operation is intended to be reversible by design (e.g., cryptography [38]) and to what

Degree, as well as if *Cooperation* of the data originator is required.

If the data originator is *Required* under the *Cooperation* attribute, the originator has to actively participate in the reversal process, i.e. the PET operation can only be reversed with the data originator's approval. Otherwise, *Cooperation* is *Not Required* and the reversal can be initiated by, e.g., a TTP. K-anonymity [99] for example cannot be reversed as long as the original data is unavailable, thus the operation requires the cooperation of the data originators. The same is true for PETs such as deniable encryption [26]: Only the data originator is able to decrypt the correct message (or a data recipient if the data has been transferred and the data recipient has been provided with the correct key). In contrast, since group signatures [30] are specifically designed to be verified by entities other than the signature creators (data originators), their cooperation in verifying a signature, or 'reversing' the signature procedure, is not required.

The *Degree* attribute describes to which extent a PET operation can be reversed and includes *Full* for fully reversible operations, *Partial* if only specific operations can be reversed or only parts of the original information can be restored, or *None* if no information can be restored whatsoever. PETs aimed at confidentiality of data records by applying encryption are obviously required to be fully reversible, otherwise their application would result in data loss. Anonymous credentials [22] can be fully or partially reversible, depending on how the credentials are to be 'opened': A partial reversal occurs when only a single pseudonym is opened and the corresponding user is identified by a TTP, whereas a global reidentification for all pseudonyms used by this user results in a full reversal.

In addition, *Deniable* specifies a special reversibility option where the data originator is able to deniably reverse an operation. This is a desirable property when the data originator is coerced to reverse the operation and disclose the original information. In this case, the data originator is able to provide data that is indistinguishable from the real original data, as is the case with deniable encryption [26].

And finally, the degree of reversibility can also simply be a non-issue, as it is the case with PETs such as PIR [31].

2.4 Pseudonymization

Generally, pseudonymization approaches have the following properties in common:

Identity focus Pseudonymization is focused on the identity aspect of privacy, i.e. aims to hide the identity of participating persons. These persons are either data originators or data consumers, depending on whether the data in question is at-rest or in-motion. The content of data is not confidential in these scenarios (or more specifically, not confidential towards the data consumers in the PET schemes) and is required to be readable (and processable) by the data consumers. Behavior is usually a non-issue as well: In case of data-in-motion, pseudonymization masks the behavior of a particular person by unlinking multiple actions and operations from

this person. However, this is actually not achieved by hiding the behavior of the person (actions) but the person's identity.

Unlinkability The goal of pseudonymization is to provide unlinkability between individuals and their data or operations. As already mentioned, confidentiality is not focused on within pseudonymization schemes, but can still be relevant for external parties. For example, considering a pseudonymized health database for research purposes, the patients are pseudonymized and researchers granted access to their health data. Still, the health data records need to be kept confidential from any external parties other than authorized researchers. Indistinguishability of individuals also plays a role in pseudonymization, but the primary focus lies on the separation of individuals and their corresponding data.

Reversibility The main difference to anonymization, the other identity-focused PET, is that pseudonymization approaches are designed to be reversible when necessary. This can be the case when, e.g., patients need to be reidentified for further consultation or when pseudonyms are required to be traced back to their holders in case of disputes. In any case, reversing the pseudonymization process is only done under specific and controlled circumstances by specifically authorized parties. Technically, reversibility is realized by protected and access-controlled ID/pseudonym-mapping information (mapping tables) or reversible algorithms requiring secret information such as cryptographic algorithms.

In this thesis, the focus lies on pseudonymization of medical data (data-at-rest) where the identities of the data originators (patients) are protected and kept hidden from the data consumers. In the following, several pseudonymization approaches for handling medical data are described.

Data-centric pseudonymization approaches mainly deal with medical data for secondary use: One of the earlier approaches developed by Pommerening and Reng ([85], [86]) relies on the combination of hashing and encryption techniques for data transport. The pseudonymization service is provided by a trusted third party replacing the unique patient identifier with a pseudonym that is derived by encryption; the encryption itself is based on a centralized secret key. An important prerequisite for this scenario is that the patient identifier is not publicly available. The final model involves multiple secondary users in a research network where a central research database is introduced containing the medical data and the unique patient identifier (again generated by a trusted third party service), and where each secondary user accesses the medical data pseudonymized by the pseudonymization service.

Bouzelat et al. [17] propose a simple anonymization protocol for exporting medical files which is actually a pseudonymization approach as it allows relinking of medical records by a trusted party. It is based on one-way hashing to replace patient identifiers so that the records can be published without compromising the patients' privacy while still allowing the linkage of multiple records to the same patient (if the patient's identifier is known) by the Department of Medical Informations (DIM) to carry out an epidemiological

study. To circumvent the susceptibility of hash algorithms to dictionary attacks, the authors propose the application of a large random file as padding information (k1). This pad is securely transferred to the data providers (laboratories) via asymmetric encryption and effectively prevents dictionary-style attacks by data recipients. Upon receipt of the padded and hashed nominal identifiers by the DIM, the data is further hashed, but with a second pad (k2), while the received data is destroyed. The twice hashed data is then published.

The approach developed by Peterson [82] involves the use of multiple encryption keys and three data tables to make personal medical data available without compromising the patient's privacy. During registration, the patient is issued a unique global key (GK) and a server side key (SSID) on a smart card. Furthermore, the patient has to provide a unique personal encryption key (PEK) and a password stored in the user table. The security table contains the reference to the user (SSID), the server-side encryption key (SSEK) and the reference to the personal data in the personal data table. The personal data is encrypted two times with the PEK and the SSEK. Data retrieval requires the knowledge of either the PEK or the GK transferred to the server that looks up the referenced SSID. With the SSID, the personal data record can be retrieved and decrypted with the corresponding keys. Data addition, deletion, and modification require the knowledge of the password in addition to the PEK or GK. As a fallback mechanism in case of a lost smart card, the patient logs in with the PEK and password and is issued a new GK.

Kalam et al. [64] use a unique patient identifier, which must not be publicly known, for deriving an anonymized identifier stored on a smart card. Furthermore, a unique project identifier is required as input where the anonymized identifier is the result of a one-way hashing procedure with the concatenation of the unique patient identifier stored on the smart card and the unique project identifier as input. That way, the patient has to explicitly consent to each secondary use of the data in each individual project. In order to protect against attacks where an external attacker tries to link data held by two different hospitals by knowing the fingerprint (hash value) of a certain patient/project and gaining unauthorized access to the database, the anonymized identifier is encrypted with a secret key only known to the hospital where the database is located. The corresponding decryption key is only known to the project participants.

Thielscher et al. [111] developed a system consisting of two databases for the patient's personal identification information and for the anamnesis data. While the datasets are stored decoupled, the relation between the patients and medical records can be restored with secret keys stored on smart cards as secure keystore. These secret keys generate unique data identification codes which are also stored in the database and do not contain any patient-identifying information. Authorizations are realized by sharing these codes between patient and health care providers, while the authorizations are only valid for a certain period of time. As a fallback mechanism in case a patient loses the smart card, a centralized patient-pseudonym list is maintained (otherwise there would be no way to recover the identifier). Since this centralized patient-pseudonym list may be the target of intrusion attacks, it is operated off-line.

Claerhout and DeMoor [33] address different pseudonym application and creation types with respect to project requirements like always using the same pseudonym for a given identifier or using different pseudonyms, creating time-dependent, location-dependent, or content-dependent pseudonyms. As for pseudonym implementation techniques, the authors distinguish between two different approaches: In batch data collection, a document register with pseudonymized (medical) payload data is used where the identifiers are replaced with pseudo identities. Pseudonymization is conducted in a two-stage process: At first, at the health professionals' local databases, the identifiers are identified and a pre-pseudonymization takes place where the identifiers are replaced with pre-pseudo-IDs. Secured by public key encryption, the pre-pseudo-IDs and payload data are then transferred to a trusted third party, where the pre-pseudo-IDs are replaced with the 'real' pseudo identities. The second approach presents an interactive data storage where there is no explicit need for local storages; there is only a single central pseudonymous database for concurrent nominative (primary) and pseudonymized (secondary) use.

Zhang et al. [129] discuss the application of pseudonyms in privacy-preserving identity management in a HealthGrid, an administrative data repository containing health-related documents generated at different points of care, similar to electronic health records. In their proposal, the authors rely on a trusted third party or a dedicated trusted HealthGrid manager for creating different pseudonyms for the same patient when treated by different health care providers using a Linkable Identity Privacy Algorithm (LIPA). A unique HealthGridID is used to internally index all records related to the particular patient, while the LIPA generates externally used pseudonyms for each corresponding health care provider, containing the HealthGridID, a random number, and a hash value calculated with a master key (only known to the HealthGrid manager or trusted third party), the health care provider's name and address, and a timestamp when the pseudonym is issued as input.

Nourmeir et al. [79] describe the pseudonymization of radiology data encoded as DICOM (cf. [77]) files for secondary use. The unique patient identification numbers in the DICOM images are replaced with pseudonyms and the files stored in a separate research database. Thereby, the authors distinguish between two kinds of pseudonyms: irreversible one-way pseudonyms (i.e. anonymization) and reversible pseudonyms. One-way pseudonyms are generated by hashing the patient identification number with a hashing algorithm. As hashing is prone to collisions, the authors propose the inclusion of the medical history in the hashing process. To prevent dictionary attacks, salting should be applied. Alternatively, a message authentication code-based hashing technique requiring a secret key may be used. Reversible pseudonyms also contain a secret key for the encryption of the patient identification number.

Iacono [60] proposes a system for multi-centric pseudonymization of health data for secondary use where a patient always has the same pseudonym, regardless of the study he/she participates in, thus allowing to link the records to the same entity, but still without disclosing this entity's identity. This approach relies on a trusted third party to create the global unambiguous inter-clinic pseudonym, but does not require any personal identifying information of the patient. This global pseudonym is calculated

using elements from all different study centers (public keys) as well as local pseudonyms created at one of the study centers. To reidentify the particular patient from the global pseudonym, pseudonym mappings are used where the local pseudonyms are stored at the individual study centers while the global mappings are stored at the trusted third party for reidentification.

Galindo and Verheul [51] apply pseudonymization to the privacy-preserving sharing of microdata allowing researchers to correlate data. Again a trusted third party is responsible for pseudonymization and is the only entity able to reidentify the individuals. In this scheme, so-called Accumulators are acting as mediators receiving data from Suppliers and forwarding it to Researchers. The Accumulators are assigned a secret symmetric key by the trusted third party, which is unknown to them. Supplying new data works as follows: the Supplier sends the (de-identified) data blocks directly to the Accumulator and the corresponding identifiers in the same order to the trusted third party. The trusted third party encrypts these identifiers with the Accumulator's symmetric key as pseudonyms and forwards them to the Accumulator where the pseudonyms are linked to the data blocks. Join and intersection operations between Accumulators also involve the trusted third party where the data blocks are sent directly from Accumulator A to Accumulator B, while the corresponding pseudonyms are decrypted and then re-encrypted with the Accumulators' symmetric keys.

Elger et al. [43] describe technical, practical, legal and ethical aspects of secondary use in the multi-institutional @neurIST research project which also includes pseudonymization. Their pseudonymization approach involves reversible pseudonyms that are created by encrypting patient IDs with secret symmetric keys (AES) unique for each clinic participating in the study. To ensure pseudonym integrity, the scheme also includes a hash value of the pseudonym calculated by a keyed hash algorithm, again with a secret key unique for each clinic. Similar to [33], a multi-stage pseudonymization with a local and a global part is used to limit the need for passing any real patient identifiers several times when a new pseudonym is required; while the local part includes the real identifiers, the global part is executed by a trusted third party only using the pre-pseudonyms.

Rahim et al. [88] also rely on a trusted dedicated pseudonymization server which is situated between the original (local) databases and the data consumers. Similar to [51], any access operation is routed through the pseudonymization server where the data is first pre-pseudonymized at the source and then transferred to the pseudonymization server via secured channels as follows: The pre-pseudonyms are encrypted with the third party's public key, while the actual medical payload data is encrypted with the public key of the data recipient (data register) to limit data disclosure at the trusted third party. The authors argue that the trusted third party not only provides optimal security against malicious attacks during the pseudonymization approach, but it is also able to dynamically control data access by monitoring the data traffic. The pseudonymization at the trusted third party should be conducted using cryptographic algorithms.

Aamot et al. [1] propose an asymmetric encryption-based pseudonymization mechanism separating actual pseudonymization and de-pseudonymization so that a pseudonymization service provider enables a specially authorized ombudsman to relink a pseudonym to

a particular patient identifier: At first, given a certain patient identifier, the pseudonymization service provider deterministically calculates a one-way mapping information (DOWMAP) and checks if it is already stored, i.e. checks if a pseudonym already exists for the patient identifier. If not, a new pseudonym is generated, but unlike other approaches, the pseudonym is not directly derived from the patient identifier. The patient identifier is then encrypted with each of the ombudsman's public keys and stored along with the pseudonym so that each of the ombudsman is able to decrypt the patient identifier for a given pseudonym without the pseudonymization service provider's participation.

CHAPTER 3

Development of a holistic pseudonym-based security methodology

This chapter describes the design of a pseudonym-based security methodology for the concurrent primary and secondary use of sensitive data (health data). The requirements are derived from the limitations of existing pseudonymization approaches described in the previous section including security aspects such as authorization and access control. The pseudonymization approach is then applied to different application scenarios. The content of this chapter has been published in the following publications:

- Thomas Neubauer, Johannes Heurix: A methodology for the pseudonymization of medical data, International Journal of Medical Informatics, Vol. 80, No. 3, pp. 190-204, 2011
- Johannes Heurix, Michael Karlinger, Michael Schrefl, Thomas Neubauer: A hybrid approach integrating encryption and pseudonymization for protecting electronic health records, Proceedings of the 8th IASTED International Conference on Biomedical Engineering, pp. 117-124, 2011
- Johannes Heurix, Michael Karlinger, Thomas Neubauer: Pseudonymization with metadata encryption for privacy-preserving searchable documents, Proceedings of the 45th Hawaii International Conference on System Science (HICSS), pp. 3011-3020, 2012
- (invited follow-up journal publication) Johannes Heurix, Michael Karlinger, Thomas Nebauer: PERiMETER - Pseudonymization and pERsonal METadata EncRyption for privacy-preserving searchable documents, Health Systems, Vol. 1, No. 1, pp. 46-57, 2012

3.1 Requirements

An analysis of existing pseudonymization approaches indicates the following limitations: The majority of the pseudonymization methods in health care is solely focused on secondary use of health data and completely neglects the possibility of using the data concurrently for primary use, i.e. in direct health care. As a consequence, data has to be stored in two states, in the non-modified original version as well as in the pseudonymized version for secondary use. Concurrent primary use would require permanent de-pseudonymized access for authorized (and only authorized) persons such as the patient and any patient-treating health professional, which in turn would require a dedicated authentication, authorization, and access control scheme. Due to the focus on secondary use, these security measures are usually not specifically addressed and simply assumed to be given. There are examples of the application of pseudonymization in primary use cases where patients' identities are protected when consuming health services by using asymmetric keypairs as pseudonyms combined with anonymous certificates [7] or where authentication and service consumption are separated and the latter is authorized by proving the validity of prepared tokens as pseudonyms [122], in essence creating a pseudonym-based security infrastructure. However, as these approaches apply pseudonymization in order to protect the patients as service consumers and not as data originators, they cannot be applied to support secondary use.

Another limitation that many existing methods have in common is their reliance on trusted third parties (e.g., [85], [33], [60], [51], [43], [88], [1]). Strictly speaking, trusted third parties are tools to circumvent security- and privacy-related problems, but they cannot actually solve the problems. Since trusted third parties have to ensure the integrity of their services, they have to implement security safeguards in order to protect their infrastructures. But introducing another security-relevant 'environment' also increases the chances of security flaws and attack vectors. Therefore, in general, it is desirable to reduce the required trust towards other parties and thus the reliance on other parties to a minimum to prevent additional loopholes and potential attackers. A prominent example of a trusted third party is a certificate authority (CA) that ensures the validity and authenticity of digital certificates (public key). Although widely accepted and regarded as a cornerstone of modern day cryptography, it has been shown that certificate authorities are also prone to attacks involving fraudulent certificates for top internet companies leading to man-in-the-middle attacks [128]. Thus, alternative approaches of decentralizing trust such as PGP's web of trust [21] have been proposed to eliminate the need for a central trusted third party, but in turn suffering from other problems like the revocation of certificates.

A further limitation is the lack of consideration for internal attackers. Security reports like Verizon's annual Data Breach Investigations Reports[1] indicate that the primary threat does not emanate from external attackers but from the inside, ranging from disgruntled employees with malicious intent to faithful employees that simply make mistakes (social engineering, improper use of privileges). While the latter can be

[1] http://www.verizonenterprise.com/at/DBIR/

mitigated by measures including awareness trainings and proper privilege management, the former poses serious problems since these internal attackers do not need to intrude into a system, they are already inside the system. With extensive access privileges, a malicious administrator is the most dangerous attacker possible. Therefore, pseudonymization can be broken if critical elements can be examined and tampered with, including offline pseudonym/ID-linking-tables (cf. [111], [60]) or cryptographic keys (cf. [82]).

Finally, the way of how to create the pseudonyms can also lead to privacy flaws. Pseudonyms are often created by some kind of one-way derivation methods such as hashing using the primary identifiers like names and patient IDs (e.g., [79]). However, this may pose a problem since these derivation mechanisms need to be deterministic in nature. To prevent brute-forcing attacks, the derivation methods thus need to include secret components such as salts or secret keys which in turn need to be properly protected and accessible by authorized parties only. The design of the derivation method also has to ensure high-entropic results to prevent statistical attacks.

Therefore, we define the following general requirements for our pseudonymization methodology:

Concurrent primary and secondary use The first requirement is the concurrent use of the stored data in primary and secondary usage scenarios by keeping stored data always in a pseudonymized state. While secondary users (researchers) are granted access to the data without the ability to de-pseudonymize it, primary users (direct-care health professionals and patients) are granted permanent de-pseudonymized access. Keeping the data-at-rest pseudonymized also reduces the threat of internal attackers and reduces the required storage space.

User controlled To counter internal attackers, privacy has to be controlled by the data originators (i.e. patients) who should be the data owners as well. That means that (i) the data owners need to be able to control who is entitled to access which data records and that (ii) this is technically enforced by the data owners.

Cryptography and key secrecy Since practical pseudonymization requires reversibility of the pseudonymization process for specific parties only, cryptography needs to be applied to prevent unauthorized de-pseudonymization. The use of offline pseudonym/ID-tables is prone to internal attackers and thus less effective than cryptography-based mechanisms. Obviously the involved cryptographic keys need to be protected adequately.

Authentication, authorization, and access control Because of the concurrent primary use, there is a need for proper authentication, authorization, and access control mechanisms in accordance with pseudonym usage. Traditional methods such as role-based access control are too coarse and thus ill-suited; de-pseudonymized access to a patient's health data should only be granted to treating health professionals, which means that a form of discretionary access control defined by the data owner is required.

Absence of trusted third parties Finally, the pseudonymization methodology should not rely on trusted third parties to realize the security infrastructure to prevent this specific attack vector.

To meet these requirements, we introduce PERiMETER, a holistic Pseudonymization and pERsonal METadata EncRyption methodology, which is based on the pseudonymization approach PIPE [95], [93], [92], [94] and aims at dealing with the following attackers: (i) honest-but-curious data consumers including patients that are interested in other patients' data, health professionals that intend to access data of patients they are not directly involved with, and secondary users who try to reidentify the corresponding patients of health data records, (ii) internal attackers such as the database administrators, and (iii) external attackers. The methodology has the following PET properties (cf. Section 2.3):

Scenario The primary untrusted entity in our pseudonymization methodology is the *Untrusted Client* in the form of secondary users accessing the pseudonymized health data and primary users who are accessing the data they are authorized for. The secondary untrusted entity is the storage server where the data is hosted since an internal attacker at the server side can never be ruled out.

Aspect Since *Pseudonymity* is the protected aspect, the directionality in our approach is *Single* (only data originators' data is pseudonymized), the holder *Individual* (pseudonyms are not assigned to multiple entities), and the cardinality *Limited* (limited reuse of the same pseudonym). In this context, the (pseudonymized) content is regarded as uncritical (i.e. available for secondary users and thus not fully encrypted) and the behavior of data consumers is also non-critical in our scenario (no need for hiding which records are accessed by the data consumers).

Aim The primary aim is to provide *Unlinkability* between patients and their individual health records as well as between the health records. Since the data is unlinked from the patients, indistinguishability of patients is not required. Deniability and confidentiality are not required either.

Foundation Our pseudonymization approach is primarily based on standard cryptographic algorithms and thus provides *Computational Security* only. The cryptographic paradigms involved include both *Asymmetric* and *Symmetric*.

Data As we investigate data-at-rest, the data property is *Stored*.

Trusted Third Party No trusted third party is involved to allow for a generally untrusted environment (frequency is *Never*, phase and task both *None*).

Reversibility And finally, de-pseudonymization for authenticated and authorized primary users requires *Full* reversibility where the participation of data owner is *Required* (only when providing the health professional with the ability to de-pseudonymize the health records, since pseudonymization is controlled by the data owner).

This pseudonymization method is aimed at data-at-rest and requires the unambiguous identification of the system users (data providers and consumers).

3.2 Methodology

Since PERiMETER is a methodology for data-at-rest, pseudonymization is applied to the health records owned by patients as data originators and not to the data consumers (patients, health professionals, secondary users) who are accessing the records. The basic concept of the methodology revolves around the following aspects:

- Fragment the sensitive health documents into non-critical fragments and store them in cleartext in a non-hierarchical fashion. Fragmentation may be done across documents or for the content of individual documents. For example, health records including discharge summaries can be fragmented into individual incidents. This includes the separation of patient-identifying sections of the documents from the actual medical contents.

- Pseudonymize the fragments, i.e. assign each fragment a number of pseudonyms to be used as access identifiers and access authorization tickets (see below). For each user who is authorized for a particular fragment, a new pseudonym is created.

- Retain the health documents' organizational structure and associations and keep them as pseudonymization metadata (pseudonym combinations) to be used as document fragment links. Knowledge of the links between different pseudonyms results in the knowledge of the correct fragments of a particular document.

- Instead of encrypting the actual health documents, only encrypt the pseudonymization metadata with a secret cryptographic key of the particular data consumer to ensure that only those persons are able to correctly reestablish the links between the document fragments.

- Provide a pseudonymization-conforming query mechanism that allows primary users to search for specific documents. This includes the ability to identify all involving fragments. Secondary users should not be able to use this query mechanism.

The use of pseudonyms as document fragment referrers results in a pseudonym-based access control mechanism where multiple cryptographic keys are employed to protect the secret links between the pseudonyms, forming a layer-based security model. Only for eligible primary users, the security model grants access to the pseudonymization metadata, or more specifically, grants the ability to de-pseudonymize and relink the document fragments. In the following, the main three features of the PERiMETER concept are described in detail. In this scheme, patients are considered to be data owners who are in full control of their data, while health professionals or health care providers who are granted primary access to documents by a patient are denoted as authorized persons.

3.2.1 Pseudonym-based access control

Pseudonyms as document identifiers provide a form of 'traceable anonymity'. In terms of pseudonymized data, access control does not refer to the traditional definition but refers to the knowledge and the ability to reconnect certain pseudonyms with each other. If someone is authorized for a particular document, this person is able to identify which pseudonyms belong together and, thus, which fragments are part of this document. If not authorized, the person cannot identify the correct associations between the fragments. In the following, we refer to the document fragments simply as records for the remainder of this chapter. In PERiMETER, a pseudonym is randomly selected from the domain of pseudonyms $Dom(\mathcal{P})$ and not derived from any entity, and then assigned to a record as access identifier. Each pseudonym can be assigned to a single record only, while each record can be assigned multiple pseudonyms. The $Dom(\mathcal{P})$ should be sufficiently large to prevent pseudonym duplicates, e.g., 128-bit universally unique identifiers (UUIDs). In general, there are two different kinds of pseudonyms which are syntactically equal but used in a different context: Root pseudonyms are available to the data owner only and represent the main record access identifiers. Whenever new records are stored, a new root pseudonym is assigned to each individual record. The (access) authorization for a trusted authorized person is represented by a set of pseudonyms called shared pseudonyms. Authorizations are always defined for specific records, and for each specific authorization, new shared pseudonyms are created by the data owner and assigned to the records. As their name implies, shared pseudonyms are shared between data owner and authorized user. While the data owner retains the root pseudonyms as primary access identifiers, the shared pseudonyms are used as access identifiers by the authorized user. Since access authorizations are controlled by the data owner, the data owner creates the shared pseudonyms for the authorized users. Technically, access revocation by the data owner is possible by deleting the shared pseudonyms, although arguably ineffective since knowledge cannot be 'taken away' from someone. Only the data owner is able to control who is authorized for which records, as long as the authorized users act faithfully and do not pass on the authorization information to third parties.

Health care providers are not necessarily authorized for the same records by the same patient. This is illustrated in Figure 3.1 where users, pseudonyms, and data records are organized as follows: The patient has three document fragments, an identification record with all person-identifying elements (name, birth date, address, etc.) as well as two different health records A and B. In this scenario, the pseudonyms are organized into pairs, one for the identification record and one for the health record forming a 1:1 relationship. Each of the health records is assigned a single root health pseudonym (root PSN HE), while the identification record is referenced with two root identification pseudonyms (root PSN ID). Furthermore, the patient has created two authorizations for health care providers A and B. Health care provider A is granted access rights to health record A sharing a shared pseudonym pair with the patient (left), while health care provider B is granted access rights to health record B, sharing another set of shared pseudonyms (right). Thus the identification record is now referenced with four identification pseudonyms, while each of the health records is referenced with two health

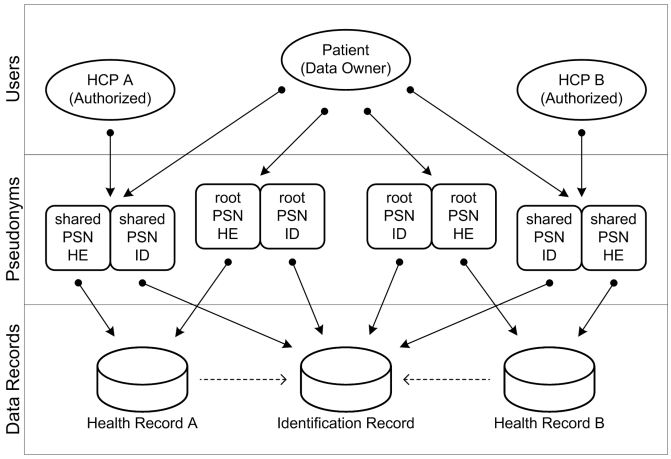


Figure 3.1: Root and shared pseudonyms organized in pairs

pseudonyms. The patient is the only person having knowledge of all elements concerning his/her identification and health records; the health care providers are aware of the records concerning their individual authorizations only. By inspecting health record B, health care provider A can only derive from the number of assigned health pseudonyms that there is an authorization for this particular record, but - lacking the knowledge of the corresponding identification pseudonyms - not that the record belongs to the patient as well. This pseudonym usage scheme results in the following: The number of randomly selected and assigned pseudonyms per record is $1 + n_{auth}$ where $0 \leq n_{auth} \leq n_{total}$ where n_{total} is the number of all potential authorizations grantees and n_{auth} the number of persons currently authorized for the particular record, and usually $n_{auth}^{rec_a} \neq n_{auth}^{rec_b}$, i.e. the number of assigned authorizations need not be the same for all records in the data owner's possession.

In order to protect the pseudonym links, pseudonyms are stored as follows: The plaintext pseudonyms are directly assigned to the records, while the pseudonym links are stored as pseudonymization metadata encrypted with user-specific cryptographic keys, i.e. pseudonyms are individually encrypted and jointly stored to represent their link. Without the key, the pseudonyms cannot be decrypted and therefore the link cannot be restored. Encryption is applied according to the layer-based security model as described in the following section.

3.2.2 Layer-based security model

PERiMETER implements a layer-based security model (cf. Figure 3.2) relying on a combination of user-specific asymmetric and symmetric cryptographic keys which protects and controls access to the data records. Here, having access to the data records means having access to de-pseudonymized documents as well. Each layer is responsible for one step in the data access process which means that the user has to pass all layers in order

to retrieve the actual de-pseudonymized documents. Technically, each layer is composed of specific cryptographic keys as follows:

- The outer layer, the *authentication layer*, consists of each user's outer asymmetric keypair (outer public key $OPuK$ and outer private key OPK) and is responsible for identifying each user unambiguously. The OPK grants access to the inner asymmetric key.
- The middle layer, the *authorization layer*, consists of each user's inner asymmetric keypair (inner public key $IPuK$ and inner private key IPK) and inner symmetric key ISK and is responsible for controlling access authorizations. The ISK is encrypted with the $IPuK$, while the IPK is encrypted with the $OPuK$.
- The inner layer, the *concealed data layer*, is the final layer and consists of the pseudonyms and the records. The ISK is used to encrypt the pseudonym links.

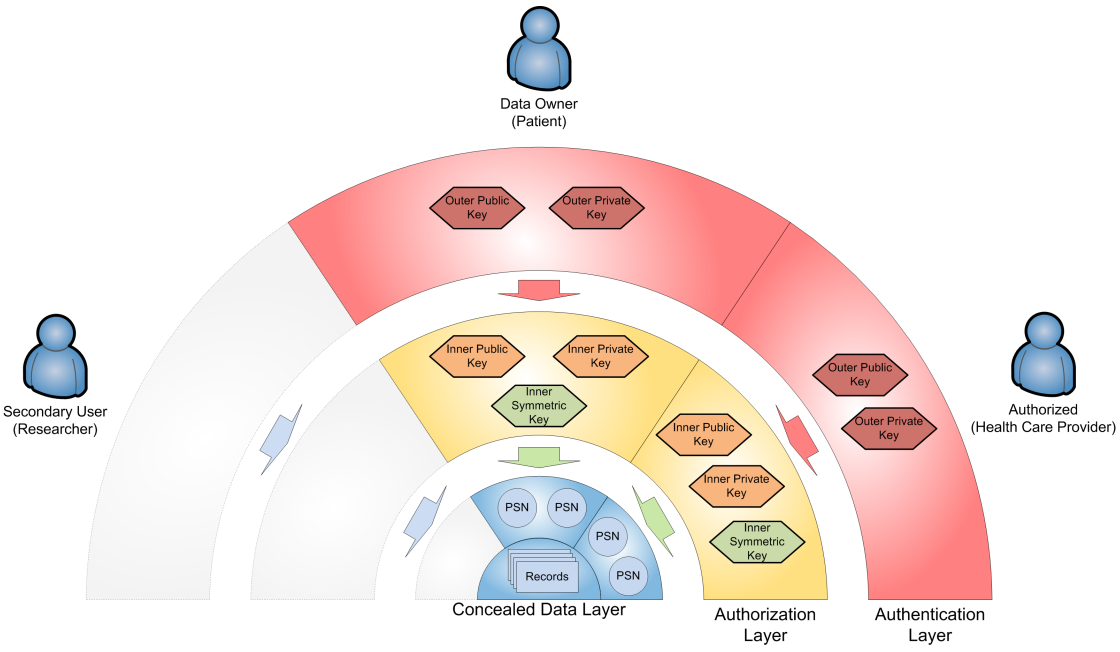


Figure 3.2: Security layers

In combination with the root and shared pseudonyms, this layer-based security model supports the following user types:

Data owner The patient as data owner is the only person who is in full control of his/her health data and can add and delete health data at his/her discretion. The data owner can grant trusted persons access to specific health records in the form of authorizations (shared pseudonym sets, cf. Section 3.2.1). Primary access to the

health records (in this example two record fragments) is achieved by using the root pseudonyms protected by the following encryption chain:

$$\{IPK_{OW}\}_{OPuK_{OW}} \tag{3.1}$$

$$\left\{\{rPSN_{R1}, rPSN_{R2}, UID_{OW}\}_{ISK_{OW}}\right\}_{IPuK_{OW}} \tag{3.2}$$

$$rPSN_{R1} \mapsto RECORD1 \tag{3.3}$$

$$rPSN_{R2} \mapsto RECORD2 \tag{3.4}$$

Data retrieval includes the following steps: First, the owner's inner private key IPK_{OW} encrypted with the outer public key $OPuK_{OW}$ is decrypted with the owner's outer private key OPK_{OW}. Then, the inner symmetric key ISK_{OW} encrypted with the inner public key $IPuK_{OW}$ is decrypted with the inner private key IPK_{OW}. Finally, the root pseudonyms $rPSN$ for records (fragments) 1 and 2 can be accessed with the inner symmetric key ISK_{OW}.

Authorized user The authorized user AU is a trusted party like a health care provider who is granted access only to specific health data of the owner. When provided with shared pseudonym sets, the authorized user is able to relink only those records referenced with the shared pseudonyms. Since both data owner and authorized user require access to the shared pseudonyms, they are encrypted with each of their inner symmetric keys:

$$\left\{\{sPSN_{R1}, sPSN_{R2}, UID_{OW}, UID_{AU}\}_{ISK_{OW}}\right\}_{IPuK_{OW}} \tag{3.5}$$

$$\left\{\{sPSN_{R1}, sPSN_{R2}, UID_{OW}, UID_{AU}\}_{ISK_{AU}}\right\}_{IPuK_{AU}} \tag{3.6}$$

$$sPSN_{R1} \mapsto RECORD1 \tag{3.7}$$

$$sPSN_{R2} \mapsto RECORD2 \tag{3.8}$$

Analogous to the data owner's data retrieval process, the authorized user requires the outer private key OPK_{AU} to get access to the inner private key IPK_{AU}, which in turn decrypts the inner symmetric key ISK_{AU} to decrypt the shared pseudonyms $sPSN_{R1}$ and $sPSN_{R2}$. With those, the corresponding records $RECORD1$ and $RECORD2$ can be identified.

Secondary user A researcher is a secondary user who has access to the records only in the pseudonymized state without the ability to decrypt the pseudonym links, lacking access to the required secret keys. Thus, the secondary user can have knowledge of the pseudonym/record mappings (since the pseudonyms are stored in plaintext here), but not of the pseudonym/pseudonym-mappings:

$$rPSN_{R1}, sPSN_{R1} \mapsto RECORD1 \tag{3.9}$$

$$rPSN_{R2}, sPSN_{R2} \mapsto RECORD2 \tag{3.10}$$

$$rPSN_{R1} \nrightarrow rPSN_{R2} \tag{3.11}$$

$$sPSN_{R1} \nrightarrow sPSN_{R2} \tag{3.12}$$

Encryption with the $OPuK$ and $IPuK$ are required to be probabilistic (with a suitable padding scheme) to prevent chosen-plaintext attacks, while ISK encryptions need to be deterministic (with a reused IV) in order to support querying by ID and pseudonyms (cf. Section 3.2.4).

3.2.3 Secure hardware and cryptographic keys

The security layers with their set of user-specific asymmetric and symmetric keys have been specifically designed to be implemented using secure hardware. In order to successfully and securely apply cryptography, three requirements must be fulfilled: (i) the security and soundness of the cryptographic algorithm must be ensured, (ii) the implementation of the cryptographic algorithm must be correct, and (iii) the cryptographic keys must be kept secret. While the soundness of modern algorithms is nowadays verified by the scientific communities and the correctness of the implementations can be checked by code reviews (in case of open source implementations, like Bouncy Castle API[2], again by a large community), the secrecy of the keys must be ensured by the cryptography operators and applications.

The majority of cryptographic implementations is software-based with keys stored in a standardized key store or container on a file system directly where the algorithms are executed or on a dedicated key management server. However, keys stored in a file system or in a database are prone to be stolen or compromised, and algorithms executed in the computer's generic memory can be tampered with, which makes software-based solutions cost-efficient but less dependable in a high-security environment. The most secure solution is a hardware-based solution where the algorithms are executed and keys are used within the secure confinement of a specially protected piece of hardware. A hardware security module (HSM) [6] is an encapsulated and tamper-resistant hardware module that is designed to withstand logical as well as physical attacks [14, 46]. The protection measures range from solid metal casings to special switches that zeroize the memory when tampering is detected [2] and are required to be certified against industry standards such as Common Criteria or FIPS 140-2. HSMs provide standardized interfaces to communicate with their host computers such as PKCS#11 [98] and serve as secure key stores and cryptographic processors which implement all standard cryptographic algorithms currently in use including RSA, ECC, and AES. They also provide secure key generation facilities and support that the keys are generated and used only within the HSM and thus never leave its secure boundaries. HSMs also come in different sizes including USB-connected devices, PCI-connected devices for installation within desktop computers and servers, as well as network-connected devices. HSMs are usually deployed in application areas where security is of utmost importance, like in ATMs for PIN management including PIN acquisition/verification/generation, in electronic payment schemes as an integral part of the back-end systems at banks processing the transactions, or in military applications as encryption and decryption modules for highly sensitive communication or as nuclear command and control tools [2]. Other applications include

[2] https://www.bouncycastle.org/

their implementation within public key infrastructures (PKI) [125] or in electronic voting schemes [97].

In addition to an HSM, PERiMETER is designed to be used with smart cards as user-side security token. A smart card is a secured (contact) micro controller card (cf. [63], [59]) that has similar properties to an HSM, i.e. tamper resistance, secured key storage area, and dedicated hardware-based cryptographic engines for common symmetric and asymmetric cryptographic algorithms such as RSA and AES. Due to these properties, a smart card is used as user-owned PIN-protected secure key store for the authentication credentials as well as a trusted client-side cryptographic engine. Since access to the smart card, or more specifically, to the keys stored on the card, is restricted by the PIN, the combination provides two-factor-authentication, an authentication mechanism superior to the simple username/password combination. For optimum security, the smart card requires a certified card reader (Common Criteria EAL3+) with an integrated keypad. The integrated keypad ensures the secured entry of the required PIN, preventing its exposure to potential malicious code installed on the host computer (malware, viruses, etc.).

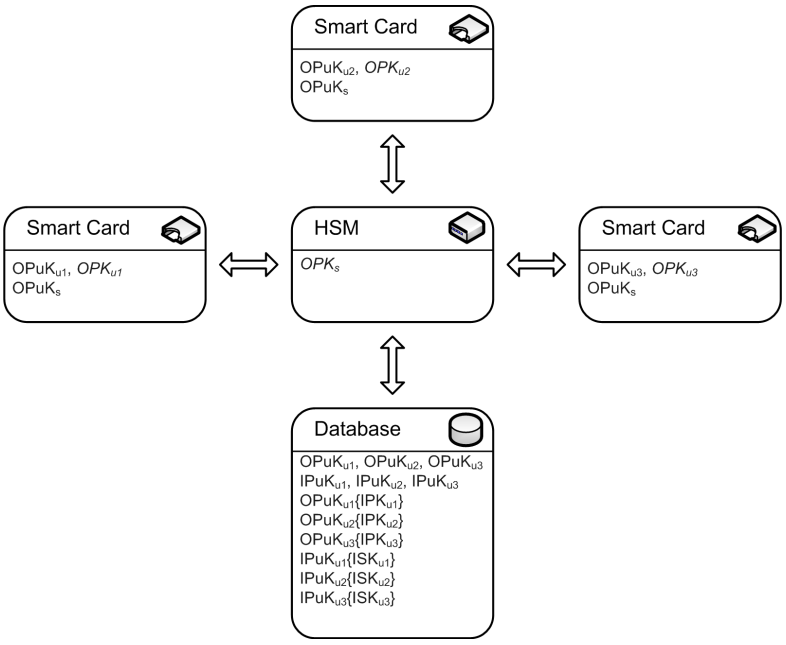


Figure 3.3: Cryptographic keys

PERiMETER aims at maximum cryptographic compatibility and does not require specific cryptographic algorithms to provide special properties like homomorphism; any general-purpose symmetric and asymmetric algorithm including 3DES, AES, RSA or ECC can be employed. There is also no restriction concerning block or key size, apart from general security considerations. Therefore, the algorithms can be easily replaced should the need arise without compromising the pseudonymization scheme. The cryptographic

keys are distributed over the user-owned smart card, a central server-side HSM and the pseudonymization database as follows to implement the layer-based security model (Figure 3.3):

- The smart card is used as tamper-resistant user authentication token and client-side secure cryptographic module for the authentication procedure. That means that the asymmetric keypair responsible for the authentication, the outer asymmetric keypair, is generated at the smart card and the OPK never leaves the card. Access to this key is protected by the PIN to realize two-factor authentication. Furthermore, the server's (or HSM's) $OPuK$ is also stored at the smart card for the authentication procedure.

- The HSM acts as tamper-resistant main cryptographic module at the server side where the health and pseudonymization metadata is stored. It uses each user's ISK for the bulk of the cryptographic operations which is imported during the user authentication phase. The ISK is only used in plaintext within the HSM. Furthermore, the HSM also stores and uses its OPK necessary for the authentication. Similar to the OPK at each smart card, the server's outer keypair is also generated within the HSM and its OPK is never available in cleartext outside the HSM.

- Finally, each user's $OPuK$-encrypted IPK and the $IPuK$-encrypted ISK are stored in the pseudonymization database. The users' public keys are also stored in the database. Since the database only stores public and encrypted secret keys, it can be considered not fully trusted; only the smart card and HSM are required to be fully trusted.

During the authentication procedure, the encrypted user keys are retrieved from the database and are decrypted either at the smart card or at the HSM. Figure 3.4 illustrates the individual steps of the authentication process including the keys involved and how they are transferred between smart card, HSM, and database. The authentication itself applies a challenge-response protocol involving the user's and server's outer asymmetric keypair and two randomly-generated nonces as user-side and server-side challenges (cf. Needham-Schroeder-Lowe protocol [68]). In Figure 3.4, we use the following notation: Subscript u and s refers to the user and server (HSM) elements (e.g., $OPuK_s$ and IPK_u), and UID refers to the user identifier. $OPuK/OPK$ and $IPuK/IPK$ are the two asymmetric keypairs and ISK the single secret symmetric key. $Gen(N)$ denotes the generation of a nonce, i.e. the random selection of a suitable nonce. $Enc(encryption\ key, \{item(s)\})$ describes the encryption of one or more *items* with the *encryption key*, while $Dec(decryption\ key, encryption\ key\,\{item(s)\})$ denotes the complementary operation of decrypting one or more $item(s)$ with the corresponding *decryption key*. $Get(user\ identifier \rightarrow key)$ refers to retrieving the *key* related to the UID. Finally, the arrows represent the messages between smart card, HSM, and database, and the operations within the three locations are shown on the lower part of the boxes. The upper part contains the keys that are stored and/or are available where

secret keys are shown in italic letters. For the sake of simplicity, the pseudonymization server which is actually responsible for the client/server and database communication has been omitted here.

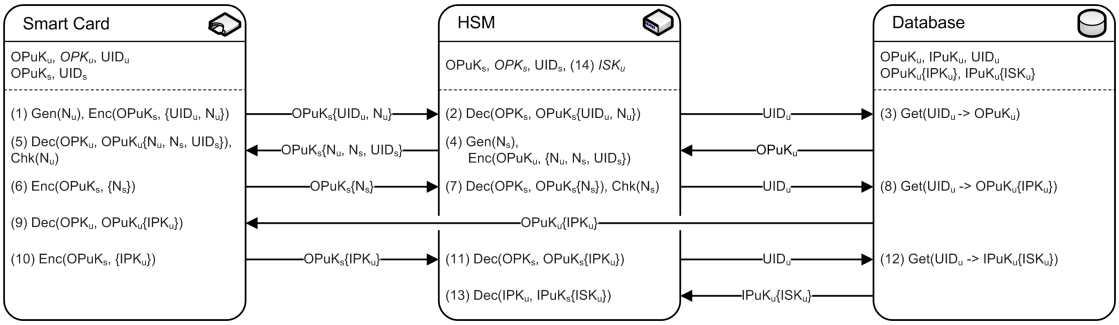


Figure 3.4: Authentication procedure with key preparation

The authentication procedure begins with the generation of the user nonce N_u and its encryption along with the user identifier UID_u with the server's outer public key $OPuK_s$ (1). Both N_u and UID_u are transferred to the HSM where the elements are decrypted with the server's outer private key OPK_s (2). With the UID_u, the corresponding outer public key $OPuK_u$ can be retrieved from the database (3). Within the HSM, the server nonce N_s is generated and encrypted with the $OPuK_u$ along with the server's identity UID_s (4). The N_u is also included as response to the user's challenge. When the smart card receives the nonces, they are decrypted with the user's outer private key OPK_u and the N_u is checked for validity (5). If the N_u is correct, the N_s is encrypted with the $OPuK_s$ as response and transferred back to the HSM (6), where it is decrypted and also checked for validity (7). If both nonces are confirmed to be correct, the actual authentication is completed and the user unambiguously identified. Next, the secret keys must be made available and loaded into the HSM. This involves the retrieval of the user's inner private key IPK_u encrypted with the $OPuK_u$ and its transfer to the smart card without an operation at the HSM (8). After its decryption with the $OPuK_u$ (9), it is re-encrypted with the $OPuK_s$ for its transfer to the HSM (10) where it is decrypted (11). Finally, the user's inner symmetric key ISK_u is retrieved from the database (12), decrypted with the IPK_u (13), and then made available for any further operations (14). This completes the authentication procedure.

In this authentication scheme, the user's IPK acts as decryption token for the ISK, which allows to keep the sensitive ISK at the server side at all times (encrypted in the database and in cleartext only within the HSM) and therefore to minimize its exposure. Since the smart card is in essence an HSM in miniature format, it can also act as primary user-side cryptographic device replacing the HSM. In this scenario, the smart card takes over all main cryptographic operations involving the ISK (pseudonym encryption/decryption) and the HSM is replaced with a conventional software-based cryptographic module. The difference in the authentication procedure compared to the HSM scenario is explained in Figure 3.5: Here the HSM is replaced with the server

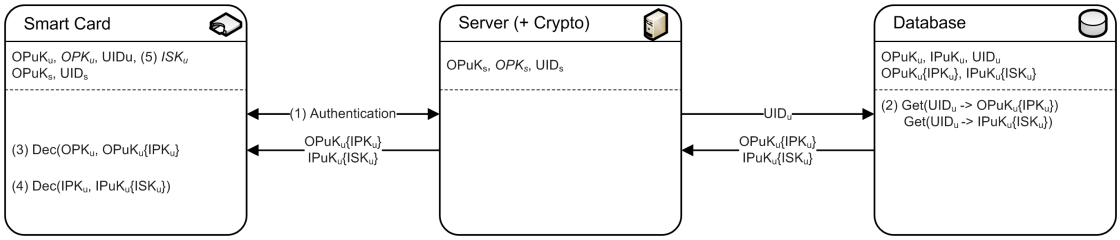


Figure 3.5: Authentication procedure without an HSM

which integrates the software cryptographic module. After the user authentication (1) which works exactly like in the HSM scenario, the server retrieves both encrypted IPK_u and ISK_u from the database and sends them to the smart card. Then, the IPK_u is decrypted, followed by the ISK_u and made available for further use (5). Since the secret keys only pass the server and are decrypted only within the smart card, they cannot be compromised without detection. Therefore, even if the user authentication without a server-side HSM would be compromised, unauthorized use of the secret keys in the database is still prevented due to the user's OPK storage location at the smart card.

As an additional security measure, a secret session key can be generated at the HSM and forwarded to the smart card during the authentication procedure to prevent eavesdropping of further communication between client and server (health documents, pseudonyms in the non-HSM scenario), realizing a TLS-like communication method.

3.2.4 Document query mechanism

Since pseudonyms must not contain any semantic information, they cannot be used for querying operations. Still, there are several ways to find the desired documents, with different levels of granularity.

The first way is to query for all documents a user owns or is authorized for by retrieving all pseudonyms that are related to either the data owner or the authorized user. This can be done due to the deterministic encryption with the ISK. Encrypting the UID with the ISK and querying the pseudonymization database yields the records for the data owner by the root pseudonym relations

$$\{UID_{OW}\}_{ISK_{OW}} \mapsto \{rPSN_{1...n}, UID_{OW}\}_{ISK_{OW}} \mapsto rPSN_{1...n} \mapsto RECORD_{1...n} \tag{3.13}$$

and for an authorized user by the shared pseudonym relations

$$\{UID_{AU}\}_{ISK_{AU}} \mapsto \{sPSN_{1...n}, UID_{OW}, UID_{AU}\}_{ISK_{AU}} \mapsto sPSN_{1...n} \mapsto RECORD_{1...n} \tag{3.14}$$

after the decryption of the root respectively shared pseudonyms where n denotes the total amount of individual records (fragments) of the owner's health documents. The same mechanism can for example be used to find all fragments k of a particular document by a single fragment x by retrieving x's root pseudonym, encrypting it, querying the database for the encrypted root pseudonym relation and thus retrieving the root pseudonyms of the remaining fragments:

$$\begin{aligned} RECORD_x \mapsto rPSN_x \mapsto \{rPSN_x\}_{ISK_{OW}} \mapsto \{rPSN_{1...k}, UID_{OW}\}_{ISK_{OW}} \mapsto \\ \mapsto rPSN_{1...k} \mapsto RECORD_{1...k} \end{aligned} \tag{3.15}$$

While this method yields either all documents or a single one, it cannot produce results based on document content. Therefore, the second method supports specific keywords that return only matching pseudonyms, more specifically, pseudonyms related to matching keywords. The idea is that keywords are stored in plaintext to support exact match as well as range queries without relying on special cryptographic properties like homomorphic encryption, while the relation to the pseudonyms are stored encrypted with the ISK (Figure 3.6). While unstructured (arbitrary) keywords are ill-suited for range queries and may also contain unwanted sensitive information, in PERiMETER only structured keywords constructed from prespecified keyword templates are used. The templates are designed for different health record properties such as document type (e.g., anamnesis, computed axial tomography), disease type, and date. Standards like the International Statistical Classification of Diseases and Related Health Problems (ICD) [127] or the Logical Observation Identifiers Names and Codes (LOINC) [89] are especially suited for the application as keyword templates.

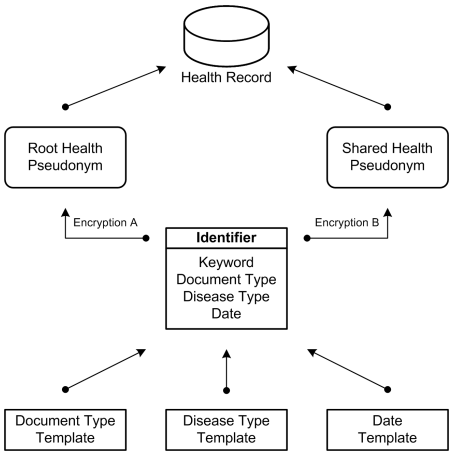


Figure 3.6: Structured keywords

A keyword template combination consisting of the document type $DocType_x$, the disease type $DisType_y$, and the date $Date_z$ is assigned a keyword identifier KID_{xyz}

which is encrypted with the user's ISK and stored along with the encrypted pseudonyms, from the viewpoint of the data owner:

$$\begin{aligned} DocType_x + DisType_y + Date_z \mapsto KID_{xyz} \mapsto \{KID_{xyz}\}_{ISK_{OW}} \mapsto \\ \mapsto \{KID_{xyz}, rPSN_{1...k}, UID_{OW}\}_{ISK_{OW}} \mapsto \\ \mapsto rPSN_{1...k} \mapsto RECORD_{1...k} \end{aligned} \tag{3.16}$$

A query for multiple keyword template combinations would yield multiple keyword identifiers, which in turn would result in multiple pseudonym relations and thus multiple health documents.

The third approach involves a personal metadata storage and is the most powerful way in terms of allowing arbitrary keywords which are stored encrypted because they may contain sensitive information. The idea is to store describing metadata (document type, disease type, arbitrary descriptions, etc.) in this personal document directory including the pseudonym references to create a personal document directory. However, this also requires a mechanism to query within the encrypted document directory. Since PERiMETER should be largely independent from the cryptographic algorithms in use, the query mechanism must not rely on specific algorithmic properties such as homomorphism. The solution is to organize the document directory in a schema-aware XML document [101] and to rely on a schema-aware labeling scheme as query mechanism [55]. Figure 3.7 shows how the personal metadata storage integrates with PERiMETER's layer-based security model (cf. Section 3.2.2): The metadata storage is part of the authorization layer and extends the pseudonym information (i.e. pseudonym relations) with description entries. To protect the metadata storage from unauthorized access, each one is encrypted with the corresponding user's ISK.

As shown in the figure, both data owner and authorized user store the same types of information, but the directories are organized differently. The authorized user's metadata storage contains the shared pseudonyms only, organized under an owner entry node, whereas the data owner's storage contains all owned root pseudonyms plus each authorization. In other words, the XML data structure allows organizing the directory entries in an hierarchical manner, as long as the structure corresponds to the predefined schema information obtained from an XML schema or DTD. Because each user has his/her own private store, each user is able to organize the document entries at his/her discretion, as long as it corresponds to the schema. To realize the schema-aware labeling scheme, each node is assigned a unique node label l which encodes the path leading to the node, as well as the node's tag name and type. For persistence, the XML document's content and structure are split and stored separately. The content of the XML document is stored at the metadata storage provider in a hash table $H(l) \mapsto E(v, ISK, n)$ where H is a cryptographic hash function, E is a symmetric encryption algorithm taking the textual value v of the leaf node labeled l as input to be encrypted with key ISK nonced with n. Here the nonce is necessary to break the determinism provided by ISK encryption since arbitrary and thus reused keywords are allowed that produce duplicate ciphers (the same XML content may exist within an XML document at different locations).

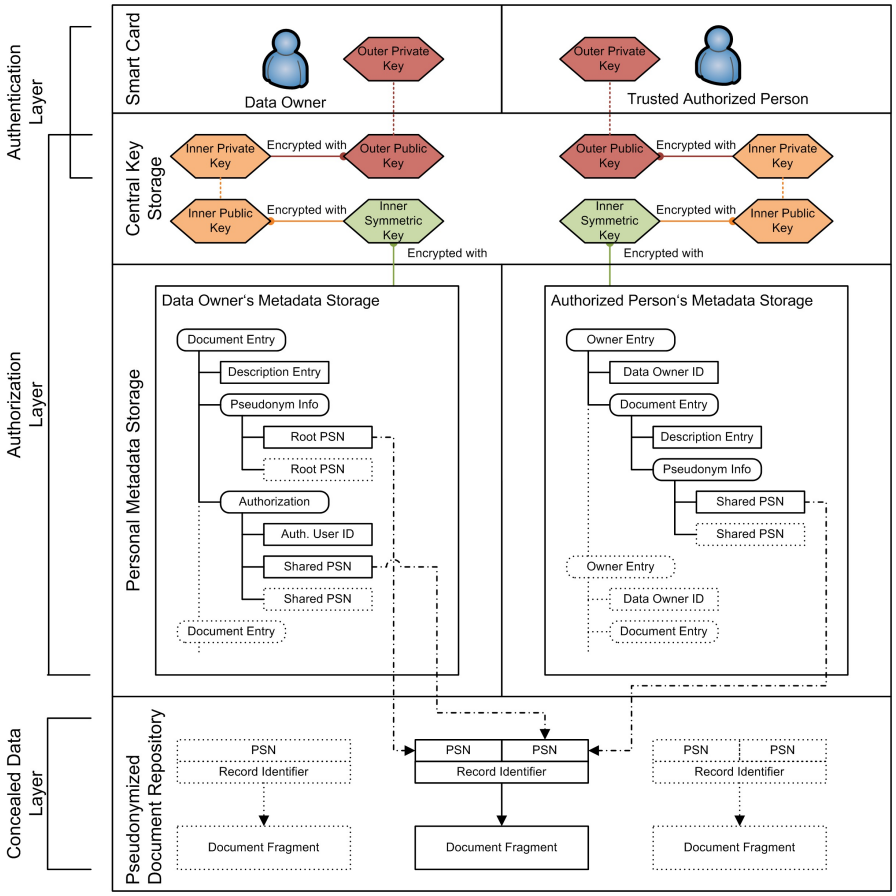


Figure 3.7: Personal metadata storage

The structure of the XML document is stored in another hash table which essentially represents a B^+-tree over the node labels in the document. To further speed up query performance, secondary index structures can be employed. To insert a new node into a document, first the node label is computed and then new entries are added to the hash tables for storing the structure and the content of the document. Queries are defined in XPath expressions. The supported XPath fragment allows for navigational queries with value-based predicates. Navigational queries are primarily processed by means of querying the structural information about the document, but also expressive node labels are exploited to reduce the number of storage access operations. To process value-based predicates, secondary index structures are used if available, or the values of nodes need to be filtered by the retriever. Basically, exact match queries, range queries, or a combination of them can be realized by defining the corresponding XPath expressions, supported by prepared secondary index structures.

Figure 3.8 illustrates this concept: The XML document consists of the text leaf nodes B and D connected by nodes A and C; only the text leaf nodes contain actual content.

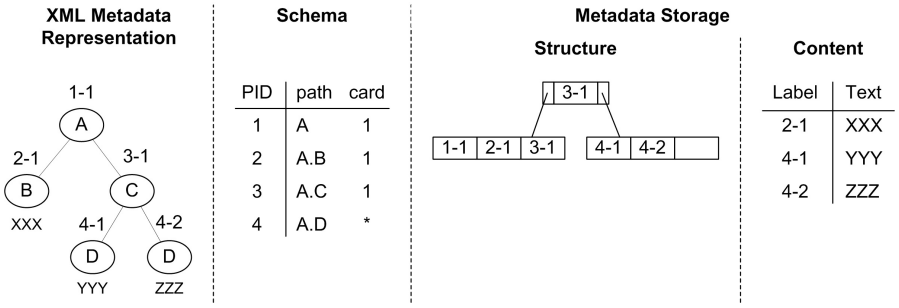


Figure 3.8: Metadata structure and content

Each node is also assigned a unique label based on its position in the XML tree (depth and number of siblings). The schema defines the path and the corresponding identifiers of each node, as well as the cardinality of the leaf nodes. In this example, node of type D has a cardinality of 2, while the schema allows a cardinality of 0 or more instances. The structure is represented as a B^+-tree that supports exact match and range queries and contains the labels of each node. And finally, the content itself is stored along with the corresponding node's label.

3.2.5 Extensions

The previous sections described the core elements of the PERiMETER methodology developed to cover the requirements as stated in Section 3.1. In the following, we describe several concepts that extend the basic methodology with useful functionality.

Affiliation

The primary authorization method using shared pseudonyms is designed to be as granular as possible to achieve a discretionary access control-like authorization scheme. This allows the patient as data owner to be in full control and limits access to a need-to-know basis. However, in certain situations it is desired to allow that these root rights are passed on to a trusted person such as a close relative. Acting as a surrogate, this trusted person should be able to exert all data owner rights including the retrieval of all documents via root pseudonyms as well as the creation of new authorizations. Technically, this can be realized by granting the trusted person access to the data owner's secret key, which we denote as affiliation. The user types as described in Section 3.2.2 are extended with the affiliate:

Affiliate The affiliate AF is a trusted person (e.g., close relative) who is granted access to the data owner's IPK_{OW} so that the affiliate is able to decrypt the owner's ISK_{OW}, which allows the affiliate to encrypt/decrypt the data owner's root and shared pseudonyms. The IPK_{OW} is encrypted with the affiliate's ISK_{AF} and stored in the pseudonymization database as affiliation token, similar to an authorization for

a health care provider where the shared pseudonyms are stored encrypted with the ISK as authorization token. The encryption chain for the affiliate is constructed as follows:

$$\{IPK_{AF}\}_{OPuK_{AF}} \tag{3.17}$$

$$\left\{\{IPK_{OW}, UID_{OW}, UID_{AF}\}_{ISK_{AF}}\right\}_{IPuK_{AF}} \tag{3.18}$$

$$\left\{\{rPSN_{R1}, rPSN_{R2}, UID_{OW}\}_{ISK_{OW}}\right\}_{IPuK_{OW}} \tag{3.19}$$

$$rPSN_{R1} \mapsto RECORD1 \tag{3.20}$$

$$rPSN_{R2} \mapsto RECORD2 \tag{3.21}$$

Before the affiliate is able to decrypt the root pseudonyms $rPSN_{R1}$ and $rPSN_{R2}$, the data owner's ISK_{OW} needs to be decrypted with the IPK_{OW}, which in turn can be decrypted with the affiliate's ISK_{AF} made available after the authentication procedure (cf. 3.2.3) involving the OPK_{AF} and the IPK_{AF}.

Since the affiliate also has access to the owner's IPK, the affiliate can also participate in asynchronous operations in place of the data owner as described in the following section.

Asynchronous operations

Due to the security layers protecting the secret ISK, all operations involving the ISKs of two persons, i.e. the creation of authorizations (shared pseudonyms) and affiliations, are basically synchronous operations where both parties have to be present to provide their security tokens so that their individual ISKs can be made available. While this represents a security feature to make sure that the correct parties are involved, this may also be hindering its practical use in certain circumstances. Affiliations are low-frequency operations and are thus less affected by the required face-to-face meeting of the participants. But for authorizations, it can be impractical to require both data owner and authorized person to be present due to the discretionary access control-style. Furthermore, since new documents are usually created/provided by the health care provider and not by the patient, an asynchronous mechanism is required to allow the health care provider as authorized person to add new documents without the patient to be authenticated at the same time. This can be realized by using the inner asymmetric keypair.

This situation may apply, for example, when a secret message needs to be forwarded to a particular recipient. In this case, using the inner asymmetric keypair has the following advantages: (i) Only the intended recipient is able to decrypt the message since the IPK is kept secret. However, due to affiliations (cf. previous section), an affiliate is also able to receive and decrypt the message, which would not be possible if the outer asymmetric keypair was used since it should be used for authentications only and thus cannot be shared. (ii) No secret key needs to be shared between message sender and receiver because each user's $IPuK$ is stored unencrypted in the pseudonymization database. And finally, (iii) because deterministic public key encryption schemes are required to incorporate randomness in the form of a suitable padding scheme (e.g., RSA-OAEP) to

prevent chosen-plaintext attacks, the same messages are guaranteed to produce different ciphertexts even when encrypting them with the same public key. This scenario of a data owner using the database as storage area for a (temporary) message for an authorized user is depicted in Figure 3.9:

At first, the data owner as message sender must be authenticated, both user identifiers UID_{OW} and UID_{AU} must be known, and the message M prepared at the server (1). Then the authorized user's $IPuK_{AU}$ is retrieved from the database using the corresponding UID_{AU} (2) and is used to encrypt the message as well as the data owner's identifier UID_{OW} as sender's identifier (3). Then, the encrypted M and UID_{OW} are both stored in the database along with the receiver's UID_{AU} which is kept in plaintext (4). This completes the owner's part. When the authorized user completes the authentication process and the server is provided with the authorized user's UID_{AU} (5), the server can query the database for any unread messages. If there is one, it is retrieved from the database (6) and decrypted using the IPK_{AU} (7) so that it is available in cleartext (8). The UID_{OW} acts as sender reference. Since the message is only temporary, it is deleted in the database after having been received (9). Depending on the message's length, it may be more efficient to encrypt the message with a generated symmetric message encryption key instead of directly with the public key and encrypt the message key with the public key (hybrid encryption).

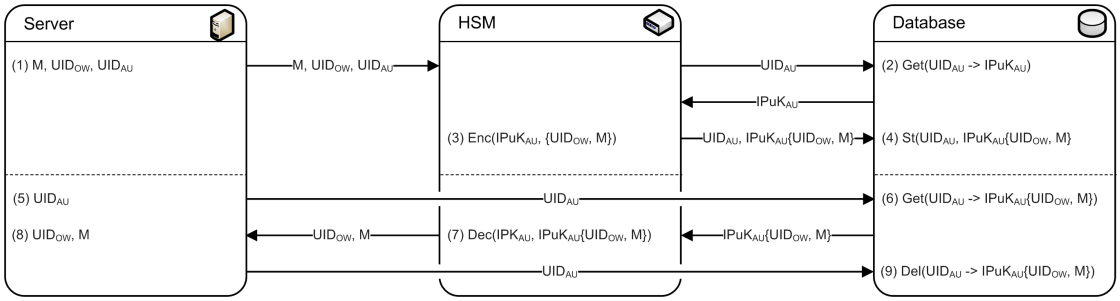


Figure 3.9: Asynchronous message

This simple message system can be extended to handle asynchronous authorizations where the pseudonym relations are stored encrypted (first with the authorized user's $IPuK_{AU}$ and then re-encrypted with the ISK_{AU} when the user is available). The data owner thus *prepares* the authorization elements which are then *accepted* by the authorized user. In particular, the asynchronous authorization involves the following steps (cf. Figure 3.10): As a precondition, all root pseudonyms $rPSN_1 ... rPSN_n$ referencing fragments of the particular medical document as well as both user identifiers UID_{OW} and UID_{AU} must be known (1). Then all records are retrieved (2) and referenced with newly generated shared pseudonyms $sPSN_1 ... sPSN_n$ (3). The new shared pseudonym/record relations are stored in the database (4). The HSM then creates the data owner's part of the authorization relation, i.e. the $sPSN_1 ... sPSN_n$ and both UID_{OW} and UID_{AU} encrypted with the ISK_{OW} (5) to be stored in the database (6). Furthermore, the owner's UID_{OW}

and pseudonyms are also encrypted with the authorized user's $IPuK_{AU}$ (8) and stored in the database as encrypted message (9), after the $IPuK_{AU}$ has been retrieved (7). This completes the authorization preparation phase for the data owner. The authorized user's phase begins after authentication when the identifier UID_{AU} (10) and all secret keys are available. At first, the database is queried for pending asynchronous authorizations. If existing, the encrypted elements are retrieved (11) and decrypted (12) in the HSM. Then, both identifiers UID_{OW} and UID_{AU} as well as the shared pseudonyms $sPSN_1...sPSN_n$ are (re)encrypted with the ISK_{AU} (13) and then stored in the database (14) as the authorized user's part of the authorization elements. Since the shared pseudonyms are now known by the authorized user, the corresponding records can be retrieved (15) and made available to the server (16) for further use (retrieving the corresponding medical record fragments). Finally, the temporary message containing the pseudonyms is deleted (17) since the authorization has been *accepted*.

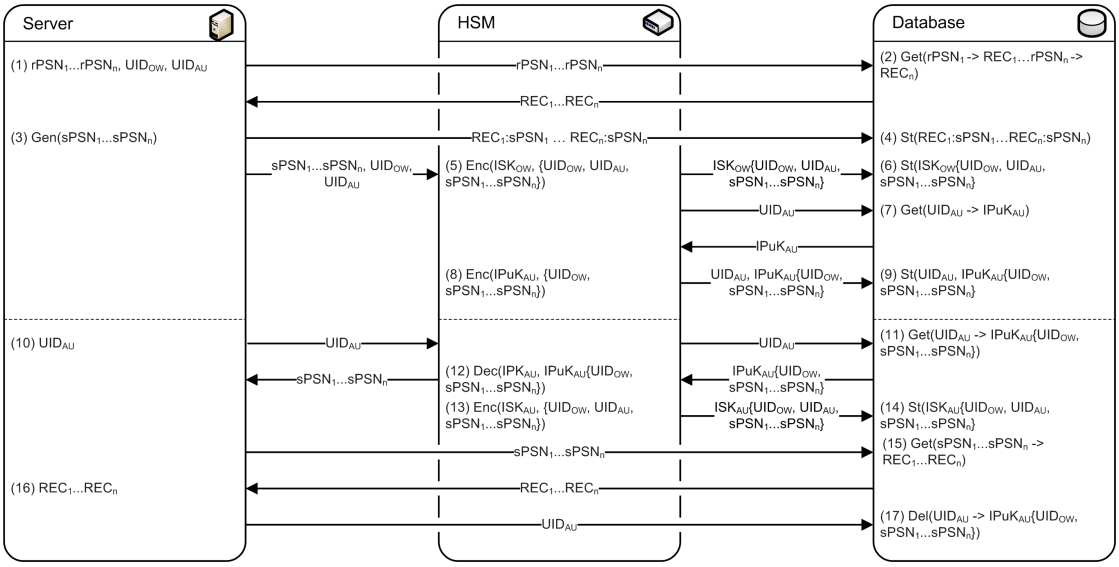


Figure 3.10: Asynchronous authorization

Another application scenario of the asynchronous operations is the storage of new medical documents which are usually provided by the health professional instead of the patient. This operation involves similar steps with roles reversed, where the authorized user *creates* the access authorization for himself/herself and the data owner *completes* it with his/her part, in addition to creating the root pseudonyms (cf. Figure 3.11): Again, the precondition is that both user identifiers UID_{OW} and UID_{AU} are known, as well as the records $REC_1...REC_n$ referring to the document fragments. The operation starts when the server generates new shared pseudonyms $sPSN_1...sPSN_n$ for the health professional as authorized user (1) and stores the pseudonym/record relations in the database (2). Then both user identifiers and the pseudonyms are encrypted with the authorized user's ISK_{AU} and also stored in the database (4). The data owner's $IPuK_{OW}$

is then retrieved (5) and the shared pseudonyms (including the UID_{AU} as reference) encrypted with the $IPuK_{OW}$ (6). To complete the authorized user's part, the (reverse) authorization is stored in the database (7). When the data owner is authenticated and the UID_{OW} as well as the secret keys are available, the (reverse) authorization is retrieved from the database (8), decrypted with the IPK_{OW} in the HSM (9), re-encrypted with the ISK_{OW} with the UID_{OW} added (10) and finally persisted in the database (11). This completes the finalization of the authorization. Since the records are new, root pseudonyms $rPSN_1...rPSN_n$ must still be generated (12), encrypted with the ISK_{OW} (13), and stored in the database (14). The root pseudonyms must also be referenced with the records, which involves their retrieval using the shared pseudonyms (15) and the storage of the root pseudonym/record relations (17). The records are stored at the server for further use (16). In the last step, the temporary authorization is deleted (18).

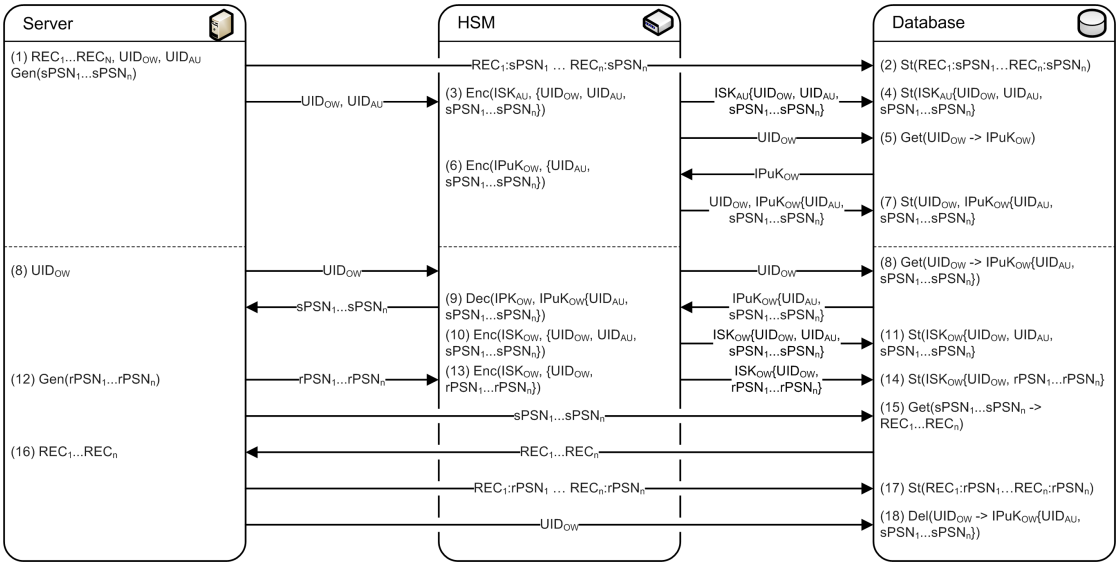


Figure 3.11: Adding a new medical document by the health professional

Before asynchronous operations can be executed, both participants, or more specifically, their user identifiers and inner public keys have to be known and authenticated to make sure that the message recipient is the correct person. Therefore, asynchronous operations require at least one prior face-to-face authorization.

Integrity

PERiMETER's pseudonymization scheme is designed to work in an honest-but-curious attacker-style scenario where the attacker is assumed to have access to the pseudonymization database. Since an honest-but-curious attacker is not able or does not intend to act maliciously, i.e. alter data without proper authorization, it is sufficient to keep the data confidential by pseudonymization. However, when considering a malicious attacker, the data's integrity will play a major role. As in general one cannot prevent

malicious data modifications in an untrusted environment, the goal is to identify if data has been altered maliciously by checking verifiable (i.e. signed) digests of data relations. Technically, the HSM with a secret key can be used to create those digests. Since the focus of PERiMETER lies on user-controlled and user-centered privacy, integrity should also be verifiable by the users. User-verifiable integrity can be realized by introducing dedicated and user-owned cryptographic integrity verification keys, a symmetric integrity verification key IVK and an asymmetric verification keypair $IVPuK$ and $IVPK$ which are used on the pseudonymization metadata (i.e. encrypted pseudonym relations and user ID/public key relations) in the following denoted as tuples. Depending on which types of tuples are required to be checked for integrity, either the symmetric or the asymmetric verification keys are used: In cases where the tuple creator and tuple verifier is the same person, the symmetric IVK is used. These include *permanent* tuples that are intended to be stored for a longer period of time, such as the user identifiers with public keys or the encrypted root and shared pseudonym relations. Here the IVK is required to be kept secret to prevent that the integrity verification digests provide information which allows the identification of the corresponding tuple creator. For tuples with different creator and verifier, the $IVPuK$ and $IVPK$ are used. These tuples are *temporary* tuples which contain temporary data elements intended to exchange information between a sender and receiver (cf. Section 3.2.5). Integrity verification is particularly important for asynchronous operations in order to authenticate the message sender.

The basis for the integrity verification of permanent tuples is the creation of the digest which involves appending the data tuple's elements (attributes) and a cryptographic hashing operation where the hash value is then signed as follows:

$$\{\{ATTRIB_1||ATTRIB_2||...||ATTRIB_n\}_{Hash}||UID\}_{IVK} \tag{3.22}$$

where $ATTRIB_x$ denotes the x'th tuple attribute, $||$ the concatenation of elements, $Hash$ the application of a cryptographic one-way hash algorithm (e.g., SHA-3), and IVK the encryption with the secret integrity verification key as the signature operation. This encrypted digest needs to be stored along with the corresponding data tuples. As usual, integrity verification is done by acquiring the IVK (stored encrypted with the user's ISK), recalculating the hash value, appending the UID, encrypting it with the IVK and then comparing the results with the encrypted stored digest, when the encryption is deterministic, or alternatively by decrypting the stored has value, decrypting it, and comparing it with the recalculated hash value in plaintext when the symmetric encryption is done probabilistically (including padding and/or different IVs).

Digest creation for temporary tuples (messages including new health documents or asynchronous authorizations) involving the $IVPuK$ and $IVPK$ follows the more traditional route of signing messages with an asymmetric technique. Here the signed digest has two functions: (i) to verify that the message's content has not been tampered with and (ii) to verify its origin (sender's identity). The digest is simply added to the tuple's encrypted content as follows:

$$UID_{recipient}, \Big\{ UID_{sender}, M, \{\{UID_{recipient}||UID_{sender}||M\}_{Hash}\}_{IVPK_{sender}} \Big\}_{IPuK_{recipient}} \tag{3.23}$$

With access to the sender's $IVPuK$ which should be proven to be valid (see public keys section below), the message's integrity and sender's identity can be verified after the message is decrypted with the recipient's IPK.

The following data tuples are required to be verifiable by signed digests:

Public keys Since PERiMETER is designed to forgo a trusted third party and thus lacks a central certificate authority, it must be assured that the public keys and the encrypted private and secret keys stored in the pseudonymization database are not tampered with. From the key owner's view, there might be the risk that an attacker replaces all inner keys, which would be possible because only the outer keys are stored at the user's security token and the user's $OPuK$ is available in the database for the HSM (cf. Figure 3.3). Even if a digest with all involved encrypted secret and public keys was created, data tampering would not be detectable as the IVK could also be faked and encrypted in turn with a faked ISK. Therefore, a secret component is required which is only known to the user and thus stored at the security token only, transferred to the HSM during authentication when the integrity of the other user keys is to be checked. This secret component $SECRET$, e.g., a sufficiently long random number, has to be included in the digest as follows:

$$\{\{OPuK||IPuK||IPK||ISK||IVPuK||IVPK||SECRET\}_{Hash}||UID\}_{IVK} \tag{3.24}$$

That way, the $SECRET$ must be correctly guessed by an attacker to produce a valid digest with the faked keys, encrypted with the faked IVK.

From an authorized user's view who needs to use a data owner's $IPuK$ for an asynchronous operation (and vice versa), the data owner's IVK must not be available to the authorized user and another mechanism is required. Since the first authorization requires both users to be logged in and authenticated at the same time[3], the data owner's public key is validated and the authorized user can create the digest as follows:

$$\{\{IPuK_{OW}||IVPuK_{OW}||UID_{OW}\}_{Hash}||UID_{AU}\}_{IVK_{AU}} \tag{3.25}$$

$$\{\{IPuK_{AU}||IVPuK_{AU}||UID_{AU}\}_{Hash}||UID_{OW}\}_{IVK_{OW}} \tag{3.26}$$

[3] An asynchronous operation is not allowed until both involved users have participated in at least one synchronous authorization operation to verify their identities.

Both users mutually sign their respective counterpart's digest in order to verify the $IPuK$ used when sending an asynchronous message, as well as the $IVPuK$ used when checking the message's integrity and origin when receiving it. Their respective IVKs are verified during their authentication operations.

An affiliate also requires an own digest for the data owner's keys because the affiliate lacks access to the $SECRET$ stored at the owner's security token. The affiliate's digest of the owner keys is constructed as follows:

$$\{\{IPuK_{OW}||IPK_{OW}||ISK_{OW}||IVK_{OW}||IVPuK_{OW}||IVPK_{OW}||UID_{OW}\}_{Hash}||UID_{AF}\}_{IVK_{AF}} \tag{3.27}$$

Again the digest is created during the affiliation process where the owner's IPK is re-encrypted with the affiliate's ISK. As the owner's IPK grants access to the owner's IVK, the affiliate can also verify, e.g., the owner's root pseudonyms' integrity (see below).

(encrypted) Pseudonyms Pseudonyms are stored in the database in tuples either in plaintext (referenced with the records) or encrypted (referenced with other pseudonyms). For root pseudonyms, only the data owner creates and verifies the digests. For shared pseudonyms, both data owner and authorized user need to have their own digests signed with their own individual IVKs (exemplary authorization for two records):

$$\left\{\{\{sPSN_{R1}, sPSN_{R2}, UID_{OW}, UID_{AU}\}_{ISK_{OW}}\}_{Hash}||UID_{OW}\right\}_{IVK_{OW}} \tag{3.28}$$

$$\left\{\{\{sPSN_{R1}, sPSN_{R2}, UID_{OW}, UID_{AU}\}_{ISK_{AU}}\}_{Hash}||UID_{AU}\right\}_{IVK_{AU}} \tag{3.29}$$

Data owner and authorized user then use their specific pseudonym tuples and digests for integrity verification purposes.

Asynchronous authorizations, new records Digests created for asynchronous authorization and new record tuples are signed with the sender's $IVPK$ and verified with the sender's $IVPuK$. The digests involve the sender's UID and the shared pseudonyms:

$$UID_{AU},$$
$$\left\{UID_{OW}, sPSN_1, ..., sPSN_n, \{\{UID_{AU}||UID_{OW}||sPSN_1||...||sPSN_n\}_{Hash}\}_{IVPK_{OW}}\right\}_{IPuK_{AU}} \tag{3.30}$$

$$UID_{OW},$$
$$\left\{UID_{AU}, sPSN_1, ..., sPSN_n, \{\{UID_{OW}||UID_{AU}||sPSN_1||...||sPSN_n\}_{Hash}\}_{IVPK_{AU}}\right\}_{IPuK_{OW}} \tag{3.31}$$

The upper relation shows the tuple for an asynchronous authorization, while the lower the tuple for a new record. In both cases, the corresponding receiver validates the tuples, removes the temporary tuples and creates the permanent pseudonym tuple(s) with the corresponding digests signed with the IVK.

Backup

The use of a user-owned security token as storage for the outer private key provides a high-level of security: The outer asymmetric keypair is generated within the secured confinements of the security token and the outer private key actually never leaves the token. Access to the key is also granted only when the secret PIN has been provided, in combination with the token itself proving two-factor authentication. However, relying on a physical device as authentication token has the disadvantage that there is a risk of the token to be damaged or lost. Whereas a new token can always be issued with a new asymmetric keypair, lacking the original outer private key, the critical inner private key can no longer be decrypted. Therefore, a suitable fallback mechanism in the form of a backup of the inner private key is necessary. The challenge here is to protect the key backup against potential attackers.

One solution to this problem is a key escrow arrangement where a trusted user is entrusted with the inner private key. Technically, the affiliation procedure (cf. Section 3.2.5) is a key escrow scheme where the affiliate is provided with the data owner's inner private key to allow the affiliate access to the owner's data, as well as to ensure a recovery option in case of an owner security token being unavailable. Since the owner's inner private key is protected by the affiliate's inner symmetric key, the key cannot be restored without the affiliate's participation. Still there are certain issues with this approach: On the one hand, this obviously requires the existence of such a trusted person, yet data owners might not want to share the inner private key with a single person at all. And on the other hand, although highly unlikely, there is a chance that both owner's and affiliate's security tokens are unavailable, which in turn blocks them from accessing the pseudonymization framework, unless the affiliate has a key escrow arrangement with a user other than the data owner.

To solve both problems, the key backup can be created with an approach based on secret sharing in the form of Shamir's threshold scheme [102] where the inner private key is split into n shares where at least k shares are required with $k < n$. Unless k shares are brought together, the inner private key cannot be restored, or in other words, with l shares where $l < k$, no information on the inner private key can be gathered. This scheme has the advantages that (i) it is more robust than the affiliate's approach as $n - k$ shares can be lost without compromising the key restoration process and, (ii) if the shares are distributed among different persons, a group of share holders is required to restore the key, which *distributes the required trust* among these share holders without the need to fully trust each individual share holder.

Shamir's approach is based on polynomial interpolation in a finite field (modular arithmetic) where a polynomial $q(x)$ of degree $k-1$ is created as follows:

$$q(x) = a_0 + a_1 x + ... + a_{k-1} x^{k-1} \; mod \; p \tag{3.32}$$

where a_0 is the secret to be split encoded as number, $a_1, ..., a_{k-1}$ are coefficients randomly chosen uniformly over the integers $[0, p)$, and p a publicly known prime with $p > a_0$ and $p > n$. Shares D are calculated by evaluating n points of this polynomial at $D_1 = q(1)$, ... , $D_n = q(n)$. Only with k of the n points of the polynomial and polynomial interpolation, the polynomial, i.e. the coefficients, can be determined, including a_0, the original secret. With $k' < k$ number of shares, a polynomial can be found which covers all these k' points for any secret a'_0, thus acquiring $k' < k$ shares does not provide any additional information, resulting in perfect security.

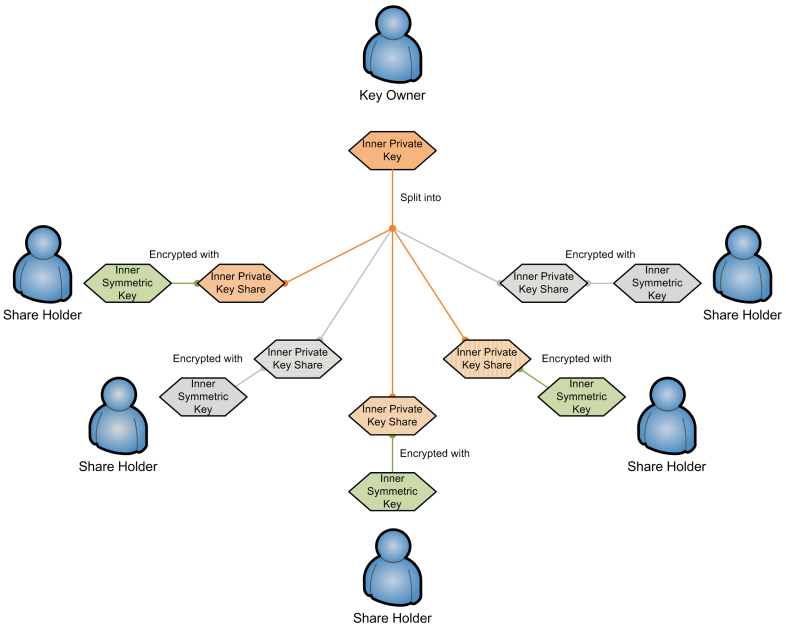


Figure 3.12: Share distribution

As shown in Figure 3.12, a user's IPK is split into five shares and distributed to five share holders. They are then encrypted with the corresponding share holder's ISK and stored in the pseudonymization database along with the user's identity. When the user's security token is damaged or lost, the user's identity is broadcast until at least three shares (orange lines) are available. Once the shares have been decrypted by the corresponding share holders, the user's IPK can be restored. A new security token is issued, a new outer asymmetric keypair is generated and the IPK encrypted with the new OPK. Along with the new $OPuK$, the encrypted IPK is then replaced in the pseudonymization database.

There are a couple of limitations with this simple threshold scheme: First, when Lagrange interpolation is used, the algorithm is limited to finite field arithmetic (modular arithmetic). Then, the share size is equal to the original secret's size, and the share distributor requires full knowledge of the secret in order to calculate the shares. There are more complex secret sharing methodologies that deal with these limitations (cf. [9] for a survey). However, for PERiMETER the limitations are no real issue: Since cryptographic keys can be simply expressed in the form of integers, modular arithmetic is perfectly suitable. Furthermore, as keys have fixed lengths, there is no sizable amount of data that needs to be produced when creating the shares. And finally, since the share distributor is the key holder (or more specifically, the security token or the HSM, depending on the secure hardware scenario, cf. Section 3.2.3), the share distributor already knows the secret IPK.

3.3 Scenarios

PERiMETER's pseudonymization methodology of splitting a document set into multiple non-critical documents (or document fragments) provides a flexible way of organizing sensitive documents, regardless of their content domain. In the following, this flexibility is demonstrated by applying PERiMETER to several document type-differing scenarios.

3.3.1 Plain documents

In the first and most generic scenario, plain documents of different types that are logically linked are organized into unlinkable documents by pseudonymization. In terms of medical records, the documents may comprise the following (Figure 3.13): A dedicated identification record contains all identifying patient information including geographic/demographic identifiers such as names, addresses, and phone numbers, as well as biometric identifiers such as size and age. The other documents contain medical data only, and each individual document does not provide sufficient information to uniquely reidentify the corresponding patient, e.g., an X-ray image of a right knee without patient identifiers or a fever chart.

In this scenario, the health records and thus the pseudonyms form a 1:n relationship, one pseudonym for the identification record and one or more pseudonyms for each medical record. From the data owner's point of view, an encrypted pseudonym relation is constructed as

$$\{rPSN_{ID}, rPSN_{HE_1}, ..., rPSN_{HE_n}, UID_{OW}\}_{ISK_{OW}} \tag{3.33}$$

where $rPSN_{ID}$ denotes the root pseudonym referenced with the identification record and $rPSN_{HE_1}...rPSN_{HE_n}$ the multiple root pseudonyms for each of the n health documents. The authorization relation contains the shared pseudonym plus the set of shared pseudonyms for only those health records the authorized user is cleared for, e.g., only for health document 1 and 2:

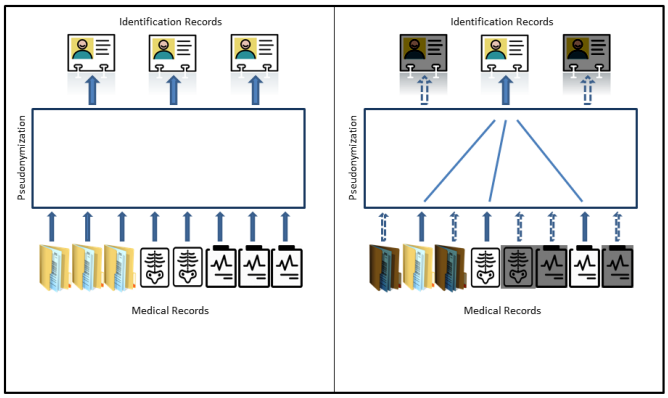


Figure 3.13: Pseudonymization of plain documents

$$\{sPSN_{ID}, sPSN_{HE_1}, sPSN_{HE_2}, UID_{OW}, UID_{AU}\}_{ISK_{AU}} \tag{3.34}$$

This structure provides a high level of flexibility, attaching medical documents to a single identification record regardless of the formers' document types.

3.3.2 Clinical Document Architecture

In addition to pseudonymization over multiple documents, the documents themselves can be split into non-critical fragments which are then individually pseudonymized. This is necessary if health documents contain sections with identifying information. To separate patient-identifying sections from purely medical content, it is helpful to rely on standardized health document structures. One example of those document standards is the HL7 Version 3 Clinical Document Architecture (CDA®) [57]. It is especially suitable for fragmentation and pseudonymization because of the following two reasons: (i) it is a widely used industry standard of representing health data in electronic health records and (ii) it separates patient-identifying information from the actual medical content.

In its current release 2, the CDA is developed by the non-profit standards developing organization Health Level Seven International (HL7) and is aimed at standardizing the structure of health documents to improve readability and interoperability between different IT systems like hospital information systems. This enables the document to be processed by different institutions regardless of how their information systems are organized and composed of. As a result, CDA documents can be exchanged electronically and displayed differently since the CDA document lays emphasis on the content of the document, not the layout. A CDA document is implemented as an XML document and consists of a highly structured header section containing person-identifying information for the patient and health professional, as well as information on the document's originator, recipient, and other administrative data, and the body section containing the actual medical content. The structure of the CDA is determined by the HL7 Version 3 Reference

Information Model (RIM) [58], an object-oriented conceptual information model that addresses general health and health care information and is the basis for all the other HL7 Version 3 standards including specifications for general patient administration or messaging.

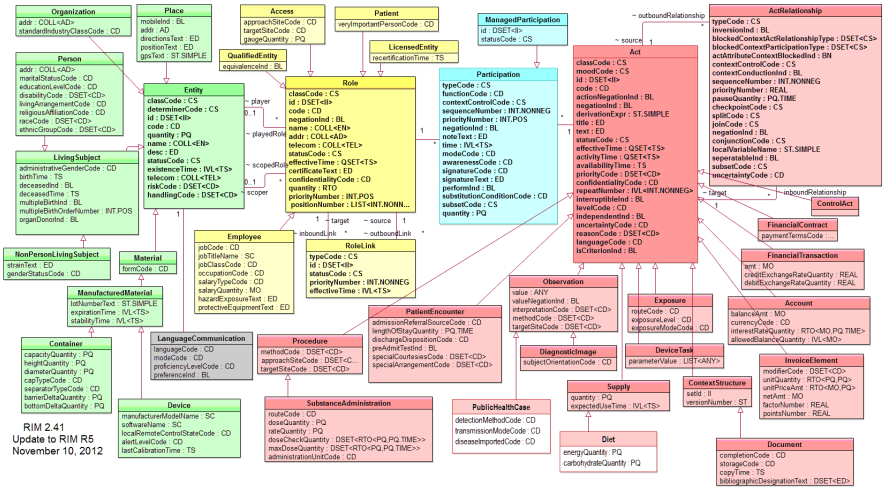


Figure 3.14: HL7 Reference Information Model version 2.41 [58]

As shown in Figure 3.14, the RIM consists of different sections representing different types of objects: entities (green), roles (yellow), and acts (red and blue). Other objects not shown in this image are elements related to messaging. CDA header elements like the patient and the involved health care professional are modeled as objects of the class *Person* of the group *Entities* exerting a specific *Role* of the group *Roles*. The health care professional is also related to the health institute modeled as *Organization*. The medical CDA body content is largely modeled as objects of the various classes collected in the *Acts* group. The RIM is further specified by multiple case scenario-specific Refined Message Information Models (R-MIMs) such as the CDA R-MIM (Figure 3.15). Due to these modeling standards, a CDA document can be easily validated by an XML Document Type Definition (DTD) file.

To ensure human-readability while also allowing automated processing, the body section of CDA documents may contain narrative text, which is useful especially for documents such as discharge letters, as well as highly-structured machine-readable codes. How 'deeply' the contents are structured is determined by three levels:

- Level 1 contains largely narrative text in free form to support the transition from paper-based documents to electronic documents. No emphasis is laid on coded information, except for the administrative header, which is defined in detail (individual sections for the patient's first and last name, home address, the health care specialist's name and organization etc.).

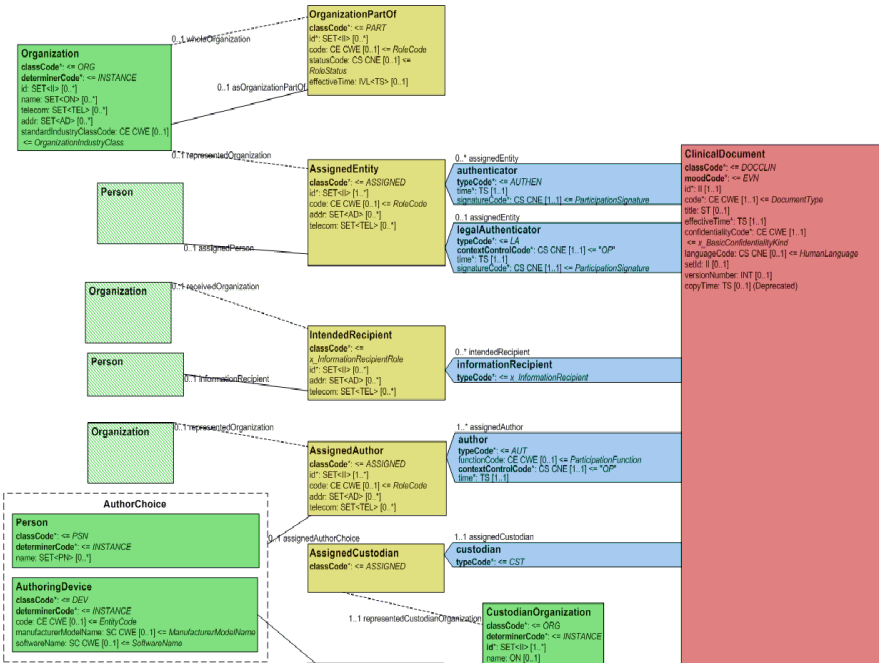


Figure 3.15: Excerpt of the CDA R-MIM specifying CDA header elements

- Level 2 extends level 1 by defining mandatory sections within the body section of CDA documents. For example, a discharge letter must contain mandatory sections including anamnesis, diagnosis, and medication so that the different medical content is organized in a more logical and semantic way. This also makes it easier to display its content more clearly in a graphical user interface of a hospital information system.

- Level 3 further extends levels 1 and 2 by including machine-readable standardized codes that are fully conforming to the RIM for optimal interoperability. For example, a level 3 CDA document's content can then be fully integrated with the HL7 Version 3 messaging system so that specific information can be extracted from the CDA document since the document has been enhanced with standardized markups to support fully automated processing. Medical incidents are encoded in standardized schemes such as Logical Observation Identifiers Names and Codes (LOINC) or Systematized Nomenclature of Medicine (SNOMED).

Regardless of its level, the content of the CDA document remains the same, only varying in its degree of machine processability.

The HL7 member countries' technical committees are responsible for creating implementation guidelines for different medical document types for different contexts. For example, HL7 Austria currently (as of 2015) provides guidelines and specifications of

clinical discharge letters for physicians and nursing personnel, lab results, medication, and radiological documents[4] in accordance with the Austrian version of a federated health information system, the Elektronische Gesundheitsakte ELGA. HL7 Germany also provides guidelines for creating more specific documents such as for the reporting of diseases for pandemic prevention. Still, due to the common basis RIM and CDA R-MIM, it is guaranteed that all these document types are readable by all systems conforming to the HL7 Version 3 standard.

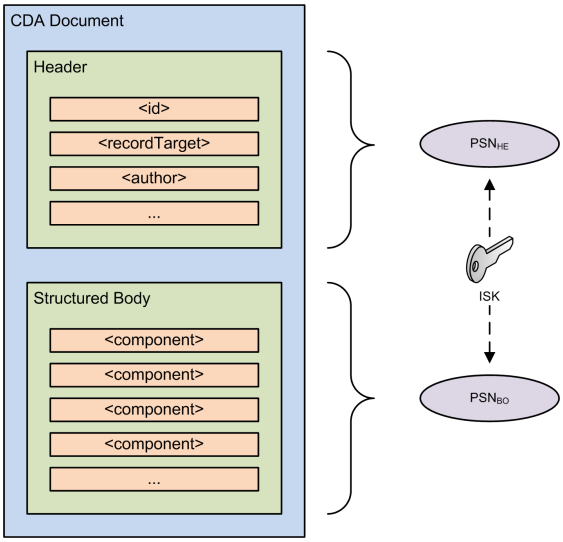


Figure 3.16: CDA document pseudonymization

Since CDA documents are already structured into administrative header and medical body sections, they can easily be split into these two parts and be pseudonymized separately. In this case, a header pseudonym PSN_{HE} and a body pseudonym PSN_{BO} form a fixed 1:1 relationship (Figure 3.16), linked with their respective sections and the relation protected by the ISK as follows:

$$\{rPSN_{HE}, rPSN_{BO}, UID_{OW}\}_{ISK_{OW}} \tag{3.35}$$

and

$$\{sPSN_{HE}, sPSN_{BO}, UID_{OW}, UID_{AU}\}_{ISK_{AU}} \tag{3.36}$$

This scenario is the basis for the conversion of real-life archived paper-based medical records into a standardized and pseudonymized form discussed in Chapter 4.

[4] www.hl7.at

3.3.3 Austrian federated health information system - ELGA

Whereas the previous scenarios focus on how to pseudonymize medical documents, the following scenario discusses the application of pseudonymization with interconnected information systems, or federated health information systems (FHIS), integrating autonomous information systems of the participating health care providers, like the Austrian implementation of an FHIS, the Elektronische Gesundheitsakte ELGA [42]. The primary goal of ELGA and FHIS in general is the facilitated exchange and processing of patient data in the form of electronic health records (EHR) to increase the quality of health care and simultaneously cut costs by improving the efficiency of health care. Not only health professionals should benefit from faster access to critical patient data, but also the patients should have facilitated access to their own data by easy-to-use public portals via the Internet to view medical documents such as radiological results. An integrated security infrastructure controlling data access and authorizations is responsible for preventing unauthorized data access.

Like the majority of FHIS, ELGA's architecture is conforming to the IHE IT Infrastructure (ITI), a technical framework developed by the Integrating the Healthcare Enterprise (IHE) initiative that has continuously been updated (currently at revision 15.0 as of late 2015 [61]). The IHE is a worldwide initiative by health care professionals and the industry to improve interoperability of health information systems with the ultimate goal of facilitating information exchange between different systems. The ITI defines so called *Integration Profiles* which define how to implement specific use cases occurring in clinical environments in a standardized way and which in part are dependent on each other. The profiles cover issues including patient lookup and administration (Patient Identifier Cross-referencing (PIX), Patient Administration Management (PAM)), access control and security (Enterprise User Authentication (EUA), Audit Trail and Node Authentication (ATNA)), patient privacy (Basic Patient Privacy Consent (BPPC)), or auxiliary profiles realizing support functions (Consistent Time (CT), Personnel White Pages (PWP)). However, we want to focus on the core of the profiles, namely the Cross-Enterprise Document Sharing (XDS) profile controlling the document exchange between different health organizations or within organizational units called *XDS Affinity Domains* [62]. The core idea of an affinity domain is to group health care enterprises using a common set of policies and sharing a common infrastructure where the patients' medical records are stored at the location of creation and not in a centralized document repository. However, to enable document sharing, searchable metadata is collected at a central document registry which allows the health institutions to identify where the required document is located to forward the document request to the correct location.

The IHE Cross-Enterprise Document Sharing-b (XDS.b) profile (cf. Figure 3.17) consists of a centralized *patient identity source* and *document registry*, and decentralized *document repositories* where the actual documents are stored and assigned a unique record identifier (RID). To enable common processing, documents are stored in agreed formats such as HL7 CDA and DICOM, and data is exchanged in compliance with IHE ITI profiles and transactions. The patient identity source contains personal data about patients and health professionals and assigns a globally unique identifier to each

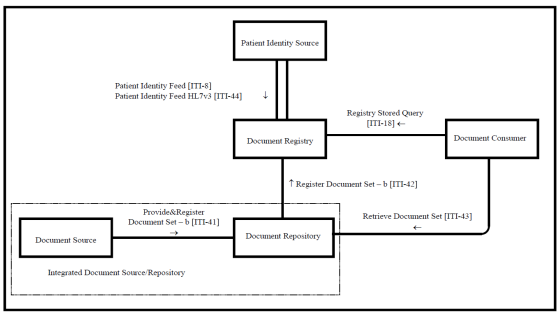


Figure 3.17: IHE ITI XDS.b Integration Profile [62]

patient (PID) and health care provider (HCP-ID). The document registry establishes the connection between a document in a document repository and the corresponding patient.

To enable search, the document source supplies metadata to the health document. For example, the metadata of a document includes the name of the health professional who created the record, the creation date and time, or the type of clinical activity. Subsequently the document is stored in the document repository and registered in the document registry using the PID, the RID generated for the document, the location of the document repository in terms of an URL, and the metadata. In case that the local storage format of the new document differs from the agreed storage format used in the document repositories, the document is transformed prior to its submission.

A document created at the source is fed into the FHIS only with the patient's consent. If the patient does not want to participate, the document is not registered at the central registry with the result that the document is retrievable only for the health professionals located at the source. Just as the patient can decide if he/she wants to participate in the cross enterprise document exchange, the patient can also control who, i.e. in addition to the document creators, is authorized to access the documents. This guarantees that patients have full control over their documents as demanded by legislation, and that it is the patients' choice with whom they want to share their medical data.

Search for documents within the FHIS is performed by (i) passing a query to the document registry, which returns the RID for each matching document together with the URL of the document repository where the document is stored, and (ii) retrieving the actual document from the specified document repository. The document registry allows to query for the documents' metadata as well as for the document owner (patient). In order to prevent information leakage, the URL and RID of a matching document are returned by the document registry only to authorized users, controlled by access and authorization profiles (BPPC, EUA, etc.).

Figure 3.18 illustrates how an ELGA area (ELGA Bereich) implemented as an affinity domain can be enhanced with pseudonymization at the document registry-level. The basic infrastructure is composed of a central patient index and a central health care provider (HCP) index, providing demographic and authentication information, including the central (global) patient and HCP identifiers (C-PID and HCP-ID). The health records

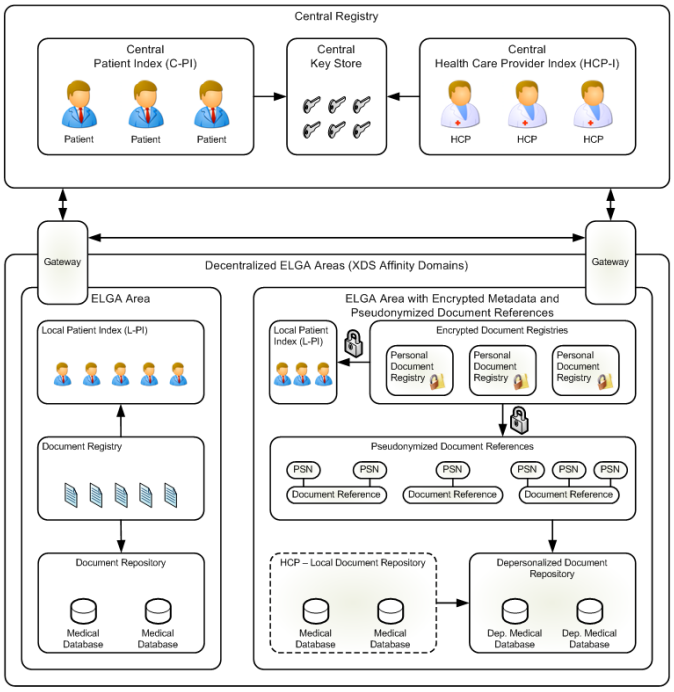


Figure 3.18: Concept of pseudonymization within an ELGA area

are managed by distributed and independent ELGA areas representing a single or a group of HCPs within the same organizational domain. Each of these affinity domains operates its own local patient index with XDS-PIDs (L-PIDs) and a local document registry containing the queryable document metadata including the references to the documents which are stored in the document repositories. Each affinity domain is connected to all other domains and to the central registry via a gateway, communicating with standardized IHE ITI-conforming transactions [62]. The core transactions include the following (cf. Figure 3.17):

- Central and local patient identifiers are queried for via transactions ITI-45 (PIX Query) and ITI-47 (Patient Demographic Query).
- Document search is realized by ITI-18 (Registry Stored Query) for local search and ITI-38 (Cross Gateway Query) for queries addressed at other affinity domains. Similarly, document retrieval is related to ITI-43 (Retrieve Document Set) and ITI-39 (Cross Gateway Retrieve).
- Providing a new health document involves ITI-41 (Provide and Register Document Set-b) and ITI-42 (Register Document Set-b).
- ITI-8 (Patient Identity Feed) and ITI-44 (Patient Identity Feed HL7v3) transactions are responsible for adding and referencing patient data to the document registry.

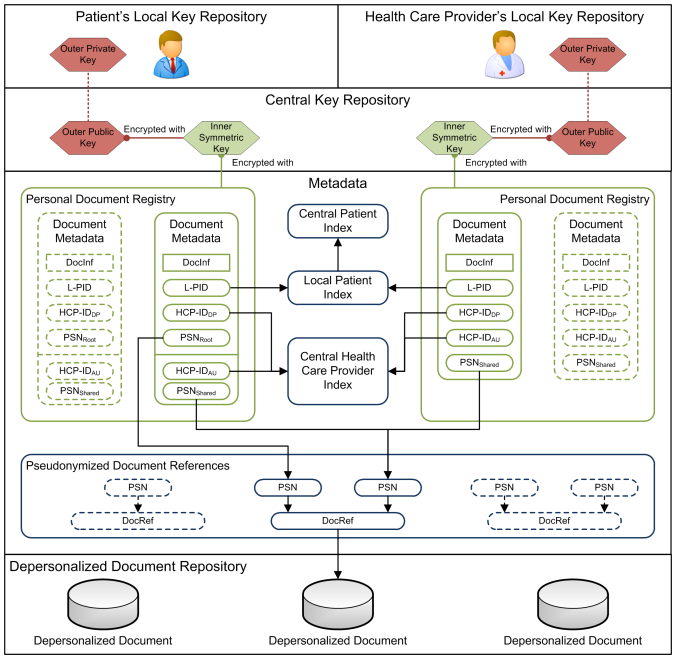


Figure 3.19: PERiMETER for ELGA

Pseudonymization within an affinity domain requires the following adaptations (Figure 3.19): A central key store is introduced keeping the patients' and HCPs' secret keys secured by encryption. While the local patient index is left unaltered, the document registry is split so that each FHIS participant maintains his/her own personal document registries, encrypted with his/her own secret key usable only after authentication with the personal security token. The document references (in case of ELGA consisting of record identifiers and locations as URLs) originally stored along with the document metadata entries are now replaced with pseudonyms which in turn are associated with the document references in cleartext in a separate pseudonymized document reference registry. Assuming that the health documents are stored in a standardized format as it is required for interoperability between different independent affinity domains, the documents are depersonalized (cf. Section 4) before they are moved from the HCP's local repository(s) to the depersonalized document repository.

Assuming pseudonymization over full documents (cf. Section 3.3.1), the encrypted personal document repositories contain the following entry for a single health document (cf. Section 3.2.2)

$$\{DocInf, L-PID, HCP-ID_{DP}, rPSN, [HCP-ID_{AU}, sPSN]^*\}_{ISK_{OW}} \quad (3.37)$$

for the patient as data owner and

$$\{DocInf, L-PID, HCP-ID_{DP}, HCP-ID_{AU}, sPSN\}_{ISK_{AU}} \quad (3.38)$$

for the authorized health professional, with $DocInf$ representing the document reference (URL), $L-PID$ the patient identifier, $HCP-ID_{DP}$ the data-providing health professional (source), $rPSN$ the root pseudonym and $[HCP-ID_{AU}, sPSN]*$ the shared pseudonym for each authorized health professional as document consumer (none, one, or more). Finally, the document reference storage contains the pseudonym/reference mappings as follows:

$$[DocRef, rPSN, [sPSN]^*] \tag{3.39}$$

To support these pseudonymization adaptations, the ITI transactions need to be modified in order to return pseudonyms instead of explicit document references. As a result, multiple lookups are required to retrieve the actual health document. In particular, the following adaptations are necessary:

- ITI-18: Reformulation of queries to suit the query mechanisms as described in Section 3.2.4.
- ITI-43: Modification to accept one or more pseudonyms instead of document references which are not transferred to the document consumer.
- ITI-39: Similar to ITI-43, modification to accept one or more pseudonyms instead of document references.
- ITI-41: Document storage in the document repository only, i.e. without automatic registration/metadata storage in the document registry (ITI-42).
- ITI-42: Modification of document registration to include pseudonym mapping storage as well as updating the personal document registries.

A data retrieval operation (either by the patient as data owner or an authorized health care provider HCP) involves the following steps (Figure 3.20):

1. The user formulates the query and sends it to the personal document registry. The query is processed and the registry returns any matching document metadata including the pseudonym in encrypted form.
2. The user selects the desired pseudonym(s) inspecting the document metadata information and sends the pseudonym(s) to the document reference storage which forwards the associated document reference(s) to the document repository to return the corresponding health record(s).

The authorization operation between patient and authorized health care provider (other than the data-providing health professional) is described as follows (Figure 3.21):

1. First, the patient executes the steps described in the previous section to retrieve the health document's root pseudonym and metadata. Then, the patient randomly selects a new shared pseudonym and transfers it to the document reference storage,

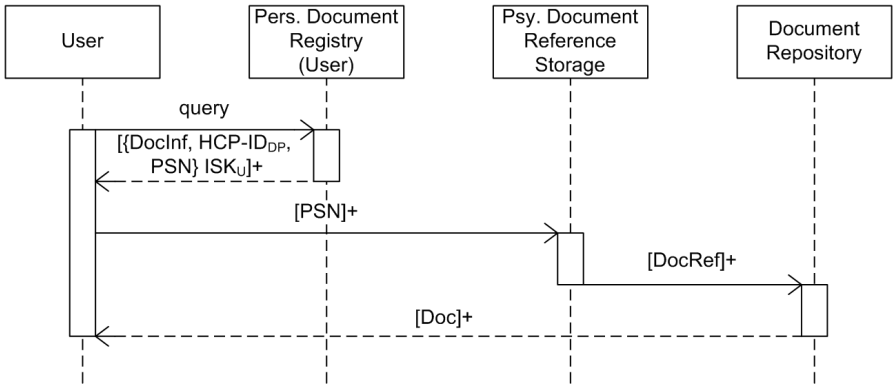


Figure 3.20: Modified ELGA data retrieval

where it is associated with the same document reference as the root pseudonym. Furthermore, the patient updates the personal document registry with the authorization information using the root pseudonym as metadata 'identifier'.

2. When both patient and HCP are present at the same machine, the document info, shared pseudonym, and patient's and HCPs' identifiers are simply re-encrypted and stored in the HCP's personal document registry. If they are not present, the patient retrieves the HCP's inner public key from the key repository via HCP-ID and sends the metadata elements to the HCP (e.g., via a shared storage area). If the HCP then logs in to the system, the authorization can be retrieved, the elements re-encrypted and appended as new document metadata entry in the HCP's personal document registry (cf. asynchronous authorization 3.2.5).

When a new health document is created and is to be registered in the central document registry, the patient has to be informed about the new document as well. Since the patient would in general not be logged in to the system at that time, this must be done asynchronously (Figure 3.22:

1. First, the document provider ($HCP - ID_{DP}$), who is also the authorized HCP in this scenario ($HCP - ID_{AU}$), copies the new document from the local HCP repository to the depersonalized ELGA document repository after removing any patient-identifying details. In addition, a randomly selected shared pseudonym is also forwarded to the document reference storage.

2. The document provider retrieves the patient's inner public key to forward the document information, the HCP and patient identifiers, and the pseudonym to the patient (via the shared storage area). The provider also stores the document metadata in the HCP's personal document registry and is automatically authorized for data access.

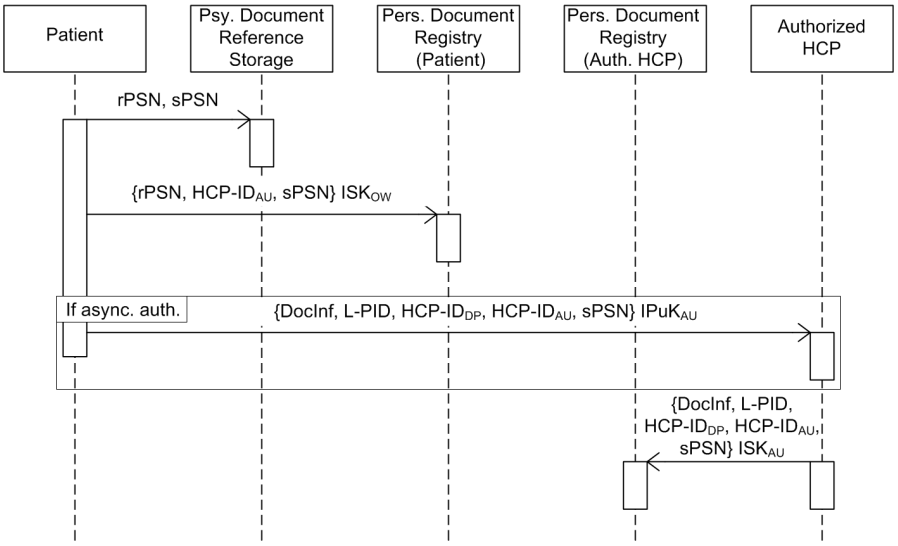


Figure 3.21: Modified ELGA access authorization

3. Upon logging in, the patient retrieves the encrypted elements and decrypts them with the inner private key. In addition, the patient also creates a new root pseudonym to be appended to the document reference (via shared pseudonym). Then the patient registers the document metadata along with the authorization information concerning the document provider in the personal document registry, concluding the document storage procedure.

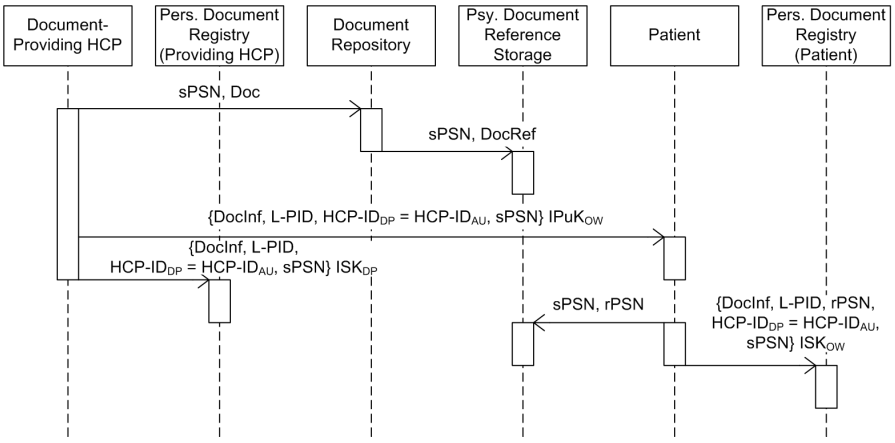


Figure 3.22: Modified ELGA document registration

Since affinity domains are designed to be independent from each other [61], the aforementioned modifications of the ELGA operations can be implemented in one ELGA

area without affecting other ELGA areas, as long as the gateway implementation remains unaffected.

3.4 Implementation and analysis

As a proof-of-concept, the PERiMETER methodology has been realized as a prototype using a combination of Java and .NET technologies for the implementation of the main logic module. As secure hardware, the prototype relies on programmable smart cards (Gemalto .NET V2+ Cards) that act as both authentication mechanism as well as client-side cryptographic processor replacing the centralized HSM (cf. Section 3.2.3); in this architecture, each user thus has his/her own client-side HSM. As storage provider, a MS SQL Server acts as key and document storage and a JBoss application server as metadata storage provider containing the record description elements, realizing all three document query mechanisms as discussed in Section 3.2.4. HL7 CDA documents are used as records to be pseudonymized (cf. Section 3.3.2) where the pseudonyms are organized in a 1:1 relationship as shown in the health document data model depicted in Figure 3.23.

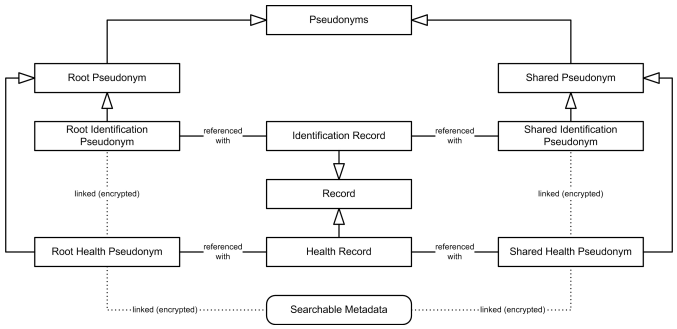


Figure 3.23: Health document data model

For each (complete) CDA health document, the corresponding metadata entry includes pseudonyms, user identifiers, and record description elements: While the data owner's metadata storage includes all root pseudonym pairs as well as all authorizations represented by shared pseudonym pairs organized under the same node, the authorized person's view involves only the shared pseudonym pair. Description elements are basically equal in both metadata databases, but can be extended with arbitrary elements by the metadata storage owner if required. The description elements are document-type specific, i.e. metadata lab results include other searchable description elements than a medical discharge letter or anamnesis.

As a precondition for the successful pseudonymization, the health record part, i.e. the CDA body section, needs to be free of any directly person-identifying information. While the majority of the personal patient-identifying information is collected in the administrative header sections of CDA documents, the body sections usually contain some

personal data as well. These Protected Health Information (PHI) elements determined by the HIPAA [116] were defined independently from the CDA standard and need to be removed from health information for proper de-identification, including names, social security numbers, telephone numbers, dates, and other potentially patient-identifying information such as account numbers and medical record numbers. Since marking and extracting these elements manually is time-consuming, automated de-identification is required to make it feasible to de-identify a large number of documents[5]. An example of a de-identified CDA body is given in Figure 3.24 where the first and last names are replaced with the generic placeholders *phi:givenName* and *phi:lastName*.

```
- <body>
  - <component xmlns="urn:hl7-org:v3">
    - <structuredBody>
      - <component>
        - <section>
            <code code="10164-2" codeSystem="2.16.840.1.113883.6.1" />
            <title>History of Present Illness</title>
            <text>*phi:givenName* *phi:lastName* visited the general
              practitioner on *phi:date* complaining of acute pain in the chest
              area. The anamnesis indicated that the patient has been a heavy
              smoker for several years and ....</text>
          </section>
        </component>
        ...
      </structuredBody>
    </component>
  </body>
```

Figure 3.24: De-identified CDA body section

The de-identified CDA body section belongs to the fictional patient Paul Jones who has recently been discharged from in-patient stay and received a discharge letter. In this exemplary scenario, he authorizes his trusted general practitioner access to the letter. For this authorization, a new shared pseudonym pair is created and assigned to the two record identifiers of both fragments. Along with record-specific description elements, the shared pseudonym pair is appended to both the corresponding root pseudonym pair in Paul Jones' metadata storage and to a newly created entry in the general practitioner's personal metadata storage. Figure 3.25 illustrates the (plaintext) document description entry from the general practitioner's viewpoint.

Apart from the shared pseudonyms and the (internal) user identifiers *iuid* for both data owner and authorized user, the entry contains general queryable elements, such as the patient's name, as well as elements unique to the *DischargeLetterDescription* type, including discharge date and a list of diagnosed diseases encoded in the ICD10 standard - I70 for Atherosclerosis. The entry also contains the PHI elements that were extracted from the CDA body section. Furthermore, the general practitioner added arbitrary private keywords including *check insurance status*. These arbitrary keywords are stored at the general practitioner's metadata storage only and are thus not visible to anyone else.

A typical query expressed in XPath has the elements

```
/child::records/child::record
```

[5] A more thorough discussion on the automated de-identification is given in the following Chapter 4.

Figure 3.25: Document description entity

```
[child::element(description, DischargeLetterDescription)]
/self::*[child::description/child::keywords
/child::keyword = 'check insurance status']
/child::element(pseudonym, SharedPseudonym)
```

and yields the shared pseudonyms of the discharge letter fragments that belong to the patient Paul Jones. With the retrieved pseudonyms, the logic then retrieves the corresponding fragments to restore the original discharge letter document. During this process, the PHI elements are extracted from the document description entity and inserted into the CDA body fragment (placeholders removed). The PHI elements' index values determined during the de-identification process ensure that the PHI elements are inserted into the correct positions.

3.4.1 Performance

Performance of PERiMETER is largely influenced by the cryptographic and database operations. While the challenge/response-based authentication procedure is required only once for each login and data retrieval is expected to be executed more often than data storage, we analyze a typical query execution and document retrieval operation in terms of required encryption/decryption and database retrieval operations. Since the

query method involving the personal metadata storage (cf. Section 3.2.4) is the most complex out of the three query mechanisms, we focus on this scenario.

A document retrieval operation can be broken down into two phases: (i) metadata query execution to identify the correct pseudonyms and (ii) the actual records' (i.e. document fragments') retrieval. The latter requires no decryption and depends simply on the number of document fragments to be retrieved. The performance of metadata query execution depends highly on the query's complexity which influences how many nodes within the XML metadata document need to be visited. The counts of hashing, database retrieval, and decryption operations are exactly $count_H = count_R = count_D = count_{visitednodes}$. Taking the query and metadata document of the previous section as an example, all *record* nodes are visited first and then checked whether their descendant description equals the *DischargeLetterDescription* and then whether one of their *keyword* nodes' values equals the *check insurance status*. If so, both *psnid* and *psnhe* nodes' values are retrieved. Depending on the number of total *record* elements within the XML document, a potentially large number of individual nodes needs to be visited to identify the requested pseudonyms.

To improve the performance of query processing, we have implemented two optimization mechanisms, namely secondary index structures and alternative node fragmentation. While the former aims at reducing the number of visited nodes to 'find' the correct entry, the latter aims at reducing the decryption operations of the actual 'values' (i.e. pseudonyms) one is interested in. For the exemplary metadata document, suitable index structures are built on *keyword* and, e.g., *dischargeDate* pointing to record. If a query contains both elements (e.g., *keyword* = *reminder* and *dischargeDate* > 2010-01-01), the results are joined to find the *record* matching both elements, thereby significantly reducing the visited nodes. If the query should always return both pseudonyms, *psnid* and *psnhe* can be stored in a single node, as well as all PHI elements, further reducing the visited nodes (and thus decryption operations). In addition to these optimizations, a typical caching mechanism is also implemented which retains the recently accessed nodes.

In terms of actual execution time, a complex query requiring tens of seconds (with ISK decryptions directly on the smart card) can be reduced to a couple of seconds with proper indexing and fragmentation, or even to far less than a second.

3.4.2 Privacy

The overall privacy assurance of PERiMETER is analyzed by investigating the following three aspects: crypto key secrecy, query secrecy, and record unlinkability.

Key secrecy

In PERiMETER, each user has his/her own set of secret keys. The outer private key always stays at the user side, while the other secret keys are stored at the central keystore protected by the security layers ($\{ISK\}_{IPuK}, \{IPK\}_{OPuK}$). Considering the tamper-resistance of smart cards and the fact that the outer private key never leaves the card, and assuming PIN secrecy, the outer private key is assumed to be reasonably secure, thus

the other secret keys as well. When no HSM is employed, the inner private and symmetric keys are decrypted at the smart card during the login procedure (authentication at the storage provider). If the card is used as main crypto module, all secret keys including the inner symmetric key are in plaintext only within the secure confinements of the card.

Query secrecy

Query secrecy depends on which query method is used. For the encrypted personal metadata method, query secrecy is ensured by the metadata storage and retrieval scheme that stores the document metadata fragmented into individual nodes represented by key/value pairs in a hash table without revealing secret information. The keys of the hash table containing the node labels (or node values for reverse lookups/secondary index structures) are calculated by a (salted) hash algorithm for efficient search, while the values of the hash table containing the node values (or node labels for reverse lookups/secondary index structures) are required to be encrypted by a (reversible) symmetric encryption scheme. A $query\left(H\left(l\right)\right)$ containing node label l masked with the hash algorithm H therefore contains no cleartext elements, while the corresponding return value $return\left(\{v\}_{ISK,n}\right)$ contains the node value v encrypted with the inner symmetric key ISK enhanced with the nonce n. Given the cryptographic strength of both the hashing and encryption algorithms and the key secrecy assumption stated above, the stored metadata in general and the queries in particular are arguably secret. The nonce prevents attacks based on frequency analysis on cipher texts.

For the other query methods by keyword or user identifier, the query reveals more information. Since these query methods are based on exact-match queries, symmetric encryption must be executed deterministically, which reveals more information than the encrypted personal metadata method [6]. In case of keywords, the query and its result reveal all encrypted pseudonyms that have the same keyword (template combination) and thus also the count of related document fragments due to the relation $\{KID\}_{ISK} \mapsto \{PSN\}_{ISK}$, though not the keyword itself. In case of query by UID, the query reveals the entirety of encrypted pseudonyms that are related to this particular UID. This is prevented in the personal metadata version due to the nonce-extended symmetric encryption, however at the expense of less-efficient query execution.

There is also a general limitation concerning PERiMETER's query secrecy depending on a potential attacker's ability to track individual queries of users: If an attacker is able to chain together multiple queries to the corresponding querier in their correct temporal order, it is possible for the attacker to identify which encrypted and non-encrypted entities belong together, potentially also revealing the relation between multiple record fragments. Potential solutions to prevent this limitation include organizational solutions such as storing documents at different locations or supplementing real queries with fake queries to obscure the real query sequence. Existing technical solutions to solve this problem using information-theoretic approaches or relying on cryptography (cf. [31])

[6] Cf. next section for a more in-depth analysis.

can be applied to individual data fragments but cannot be easily applied to (file-based) documents.

Record unlinkability

The main privacy property of PERiMETER, record unlinkability, is provided by the pseudonymized document fragments. The general goal is to mask the relation $user_x \neq rec_x$. Given that the record is properly depersonalized, this is achieved as follows: First, the direct mapping is broken up by introducing a set of individual pseudonyms so that $user_x \mapsto psn_x \mapsto rec_x$. For a second $user_y$ being authorized to access the same record rec_x, the following additional mapping exists: $user_y \mapsto psn_y$ while $(psn_x/psn_y) \mapsto rec_x$. The latter pseudonym/record mappings remain in cleartext whereas the former user/pseudonym relations are persisted encrypted with the ISKs. Therefore, $\{user_x \mapsto psn_x\}_{ISK_x}$ and $\{user_y \mapsto psn_y\}_{ISK_y}$ where due to the key secrecy assumption $user_x \mapsto user_y$ (i.e. authorization relation) or $user_x \mapsto psn_y$ cannot be identified, unless $user_x$ is the data owner and $user_y$ the trusted authorized person, in which case $user_x$ has created psn_y as authorization. As a result, all unauthorized persons cannot identify $user_x \mapsto rec_x$, and, assuming that another $user_z$ has been authorized for rec_x, both authorized persons $user_y$ and $user_z$ have the knowledge of $user_x \mapsto rec_x$, but the authorization relation $user_x \mapsto user_y$ is hidden for $user_z$ and vice versa.

Since pseudonyms are randomly selected and used only once for each record fragment, no substantial information can be gathered with frequency analysis of pseudonyms, even when deterministically encrypted with the same key. However, a static analysis of the pseudonymization metadata may reveal information due to deterministic encryption depending on the stored tuples and their storage structure. If, for example, the UID is encrypted deterministically and stored individually so that it can be used in exact-match queries (see above), an analysis of the encrypted root pseudonyms reveals the number of record fragments owned by a particular user, but it does not reveal which user. In general, deterministic encryption affects leaked information of pseudonymization metadata as follows:

Keys User keys stored at the database comprise outer and inner public keys and the encrypted private and secret keys. Since the public keys are selected randomly for each user and obviously need to be publicly known, only the encrypted private keys' secrecy has to be checked. The asymmetric encryption scheme is probabilistic (e.g., RSA with OAEP) to prevent chosen-plaintext attacks. However, since the asymmetric keypairs are chosen randomly with a key size sufficiently large (min. 2048 bit for RSA or 224 bit for ECC), the chosen-plaintext attack to identify a correct IPK or ISK is still practically an unfeasible brute-force attack. Apart from that, the number of keys also disclose the number of users in the system.

Root pseudonyms Plaintext root pseudonyms are used only once and thus exclusively tied to a single document/record. Therefore, root pseudonyms encrypted deterministically with the same ISK yield different ciphertexts. The encrypted user

identifier stored along with the encrypted root pseudonyms though discloses (i) the number of encrypted root pseudonyms and thus documents that are owned by the user and (ii) the number of document fragments that belong together.

Shared pseudonyms Plaintext shared pseudonyms have the same properties as root pseudonyms and are also exclusively tied to a single document/record. However, since shared pseudonyms represent single authorizations, a single record can be connected to multiple share pseudonyms. Each user can also have multiple authorizations, both as authorization grantor and grantee. Thus, the encrypted shared pseudonyms may indicate (i) the number of authorizations a particular user has granted and/or (ii) the number of authorizations a user has received. Furthermore, the number of plaintext pseudonyms assigned to each record indicates (iii) the number of authorizations for this record (one root and x shared pseudonyms).

Keywords Each composed keyword is assigned a unique ID which is encrypted and stored along with the encrypted root/shared pseudonyms. If a particular keyword is used multiple times by the same user resulting in the same ciphertext, counting the encrypted keyword IDs indicates the number of times the keyword is used, i.e., how many records have the same keyword, but obviously not which keyword. If the same keyword is used by multiple users and thus its ID encrypted with different ISKs, the total number cannot be deciphered due to the different ciphertexts.

Asynchronous messages By investigating the number of asynchronous messages stored in the pseudonymization database, the total count of messages as well as their recipients can be identified. However, due to probabilistic encryption involving the $IPuK$s, the message contents including the message creators' (e.g., authorization grantors') identities are kept hidden.

Digests Digests to validate the integrity of pseudonymization metadata come in two flavors: Created with a symmetric IVK when the integrity verifier is the key owner or created with a private $IVPK$ when the verifier isn't the key owner and uses the corresponding $IVPuK$. In the former case, each digest is unique since the attributes involved in hash calculation for a particular digest differ from those of another digest (user-specific unique cryptographic keys, unique root and shared pseudonyms), even if digests are encrypted deterministically with the IVKs. Digests for asynchronous messages are signed with the sender's $IPVK$ and added to the message which is encrypted with the receiver's $IPuK$. Since these digests are integrated into the encrypted messages, digests do not leak information.

If the third query mechanism with encrypted personal metadata stores (cf. Section 3.2.4) is employed, replacing the keywords as described above, the root and shared pseudonyms are organized differently and encrypted probabilistically, preventing the information leaks concerning the pseudonyms, but also reducing query efficiency.

CHAPTER 4

Extension to secondary use - a real world scenario

This chapter describes the application of the pseudonymization technique in a real-world scenario where archived paper-based health records are pseudonymized before they are made available for secondary use. Before the records can be pseudonymized, several preconditions must be fulfilled including the proper identification of patient-identifying elements (PHI) as well as the conversion of the health records into a standardized universal format (CDA). The following text describes how these requirements can be met and demonstrates the approach with a sample composed of real-life archived health records. The content of this chapter has been published in the following publications:

- Johannes Heurix, Antonio Rella, Stefan Fenz, Thomas Neubauer: Automated Transformation of Semi-Structured Text Elements, Proceedings of the 2012 Americas Conference on Information Systems (AMCIS), pp. 1-11, 2012

- (invited follow-up journal publication) Johannes Heurix, Antonio Rella, Stefan Fenz, Thomas Neubauer: A rule-based transformation system for converting semi-structured medical documents, Health and Technologies, Vol. 3, No. 1, pp 51-63, 2013

- Stefan Fenz, Johannes Heurix, Thomas Neubauer, Antonio Rella: De-identification of unstructured paper-based health records for privacy-preserving secondary use, Journal of Medical Engineering & Technology, Vol. 38, No. 5, pp. 260-268, 2014

- Johannes Heurix, Stefan Fenz, Antonio Rella, Thomas Neubauer: Recognition and pseudonymisation of medical records for secondary use, Medical & Biomedical Engineering & Computing, Vol. 54, No. 2, pp. 371-383, 2016

4.1 Requirements

In the previous chapter, the application of pseudonymization of health data has been demonstrated as an effective way to support secondary use of health data while preserving the corresponding patients' privacy. The PERiMETER approach is suitable for combining primary and secondary use of data where pseudonymization is applied to allow primary users, i.e. patients and health professionals involved in direct-care, to reidentify the patients' health records. However, pseudonymization can also be used in a pure secondary use scenario where, in general, secondary users are granted access to the pseudonymized data. In addition, these users have the option of selective de-pseudonymization controlled by an authorized person such as a data manager if the need arises. In this case, instead of the patients, the data manager is the data owner who is bound by legal regulations to ensure privacy. In particular, the following workflow sketches the application of pseudonymization in such scenario:

1. Health records in a research database for, e.g., epidemiological research, are stored in a pseudonymized state and are available to registered researchers.

2. A researcher queries the pseudonymized data records for useful entries for a study, e.g., searching for patients/records with specific diseases.

3. The researcher requires additional information like other records of the particular patients or requires the reidentification of the patients to (directly) request further information or permission to use the data.

4. The authorized person can trigger the selective de-pseudonymization if the request is valid to either identify the remaining patients' documents (but still pseudonymized) or actually reveal the patients' identities.

In order to realize such scenario while pseudonymization is in force, three requirements have to be met:

Depersonalization Real-life medical records are largely unstructured and contain many patient-identifying details, so the first step of creating documents for privacy-preserving secondary use is to scrub these identifiers from the records. While this is relatively easy for a domain expert to do for a few documents, it becomes unfeasible to do manually for a large number of documents. With highly structured documents, it is simple to remove PHI in an automated approach, but as medical records may contain large segments of narrative texts, it becomes much more difficult to identify PHI automatically. Thus, an automated method to identify PHI even within narrative text is required to depersonalize medical records.

Standardization Since medical information systems have been heterogeneous solutions which have usually grown over time, documents produced and managed by these systems also differ in structure and the way they organize the medical content,

which leads to problems with interoperability and information exchange. When documents conform to standards like HL7 CDA, it would greatly improve the interoperability and thus would allow the documents to be made available to a larger research community. The situation has somewhat improved with the introduction of the electronic health records and their standardization mechanisms driven by organizations such as the IHE and HL7. Still, a great source of information for secondary use is largely unaffected by this standardization effort due to its age, namely patient records maintained in archives by medical professionals due to legal regulations. These archives contain largely paper-based records digitized (i.e. scanned) as image files. Before these documents can be used, they must be brought into a standardized form.

Information extraction The standardization of documents requires the transformation of the documents' content by identifying, extracting and converting useful text segments and information into the new document standard. For example, a CDA discharge letter requires patient and health professional identifiers in the header section and the medical contents in the body section, separated into segments of the patient anamnesis, diagnosis, and medication (among others). Similar to depersonalization, this is simple for a limited number of documents but unfeasible to do manually for a whole archive of documents. Therefore, the information extraction process has to be executed in an automated way in order to efficiently process the large amounts of data.

Information extraction (IE) is a subdomain of the natural language processing (NLP) discipline and is tasked with the automated extraction of structured information from unstructured sources [100]. IE is closely related to information retrieval (IR), but IR's goal is to return documents as a result of a search whereas IE returns information or facts [74]. A typical IE system has two objectives: (i) to identify and annotate potentially relevant information in the narrative input text and (ii) to actually extract and transform the desired information into the target form. The annotation task usually involves multiple preprocessing steps including tokenization, part-of-speech tagging, or parsers for boundary and named entity recognition which are executed in a pipeline where the output of the former step is used as input for the next step to improve the quality of the overall result. Existing work largely does not distinguish entity recognition or annotation and actual information extraction. Text processing steps are composed into adaptable annotation frameworks such as GATE [36], C-PANKOW [32] or UIMA [47] which provide their different processing functionality like tokenization or whitespace identification through special plug-ins.

IE systems can be categorized into two fundamentally different types [4]: (i) the Knowledge Engineering Approach with hand-made rules written by domain experts to identify and correctly mark the relevant information entities, and (ii) the Automatic Training Approach where the system creates the rules itself by analyzing manually pre-labeled (annotated) training corpora. There has been an ongoing debate on which of these approaches performs better [4, 100]: Knowledge engineering-based systems tend to

produce good results very fast, especially when training data is tedious and costly to acquire. They also excel when the annotation and extraction specifications are likely to change in a foreseeable way (e.g., when the layout of documents are updated). The big downside of hand-crafted rules is the actual work to define them. Creating accurate rules often relies on an iterative testing and adapting process. Training-based systems relieve the domain expert from this manual rule-creation work which means that no (potentially) complex rule formalisms need to be learned. But in essence, automatic training-based systems shift the workload from creating the rules to annotating the training corpora. To produce reasonably good results, these systems require a large number of manually annotated documents. Thus, selecting the adequate system for a particular IE problem highly depends on the availability of training data and the structure thereof, the technical expertise of domain experts, as well as the structural and lexical properties of the document base.

IE systems have been applied in a multitude of applications including enterprise applications, scientific purposes, or web-based systems [100]. Although originally developed outside the biomedical and clinical area, information extraction has meanwhile been adapted to the medical domain as well, where its main application was to identify PHI so that they can be annotated and extracted for privacy-preserving use. In general, both rule-based and machine learning based techniques can be found, since neither approach has turned out to be clearly superior.

Earlier approaches relied on lookup tables and dictionaries, but usually incorporated other aspects like POS or other semantics as well. Taira et al. [110] developed an algorithm that uses lexical lookup tables with more than 64 000 first and last names and semantic constraints to calculate the probability of a word being a name. The algorithm scans each sentence and extracts potential names based on the structure of the sentence. While the approach works well to identify patient names, it cannot be used to identify further personal or medical information. Berman et al. [11] developed an algorithm to remove personal data from pathology reports. The algorithm steps through the text and a word matching a Unified Medical Language System term (UMLS) is replaced by the UMLS code and a synonym for the original word. High-frequency words ('stop words') such as 'a', 'an', 'the', or 'for' are left in place. All other words such as personal data are replaced by asterisks. The problem of this approach is that it removes more than just personal data, and the resulting text is hardly readable. In [12], Berman presented an improved approach which uses a list of safe words (i.e. words which do not refer to personal data) denoting those that can be kept in the document, while the others are replaced by an asterisk.

In 2006, an i2b2 natural language processing challenge focused on de-identification [119] was held. The submitted approaches were dominated by machine learning-type systems, though some used a combination of machine learning and rules. For example, Hara et al. [56] use regular expression pattern matching for identifying phone numbers, dates, and IDs. A sentence classifier based on a boosting algorithm and phrase chunker based on support vector machine (SVM) technology are used to identify ages and the name of hospitals, patients, and doctors. The features used for the SVM include headings

of sections (closest to the target), the category of the sentence as determined by the sentence classifier, part-of-speech (POS) tag, orthography, and the root and surface form of the target. Sentences are classified based on the PHI they contain. Evaluation has shown that the entire system performs better without sentence classification. Szarvas et al. [109] developed a feature set for the word-level classification model, describing the word characteristics along with its actual context (window of size four). The model does not use deep domain knowledge such as part-of-speech, chunk codes, ontologies, or domain-specific resources such as MeSH IDs. The following feature categories are used: (i) orthographic features, (ii) frequency information, (iii) phrasal information, (iv) dictionaries, and (v) contextual information. Additionally, the authors used regular expressions, a dictionary of common headings observed in typical discharge records, and trigger words/phrases to identify PHI. The Boosting algorithm and C4.5 classifiers are combined in an iterative learning approach for word/phrase classification. Wellner et al. [123] used the MITRE Carafe toolkit based on conditional random fields (CRF) to de-identify medical records. Within Carafe, the BIO representation is used to identify phrases within sentences. Additionally, the authors used location dictionaries and regular expressions to capture the more standardized PHI (e.g., dates). Although the system performs very well, 50 percent of the errors (131 in total) are due to insufficient classification, i.e. the classification system did not identify and tag PHI (e.g., location).

Still, rule-based approaches continued to emerge, like the Medical De-Identification System (MeDS) by Friedlin et al. [50] for de-identifying Health Level Seven (HL7) messages and narrative text documents. It removes identifying information in four scrubbing processes and one preprocessing process. The main strength of the system is that it does not rely on a single method or process to remove identifiers. Regular expressions, name lists, header information, and word-nearness similarity algorithms (in the case of misspelled names) are used. Geographic name databases and the integration of natural language processing techniques are mentioned as possible extensions to enhance the effectiveness of the de-identification system. Velupillai et al. [121] developed a system for de-identifying electronic patient records written in Swedish. The authors customized the de-identification software package De-id [76], a rule-based system relying on rule sets, heuristics, and supplemental dictionaries, for Swedish health records. As the system is rule-based, regular expressions and dictionaries for Swedish names, locations, and diseases were created. The main conclusion was that the American system De-id did not yield good de-identification results when used for Swedish health records (even with adapted rules and dictionaries), thus requiring a completely new Swedish de-identification system using rules, lexical resources, and semi-supervised machine learning techniques. Grouin et al. [54] came to a similar conclusion when they tried to adapt the de-identification software package De-id to the French language. Again, simply modifying the system and its rules resulted in low precision and recall.

One of the main limitations of these solutions is their focus on the identification of PHI, whereas the information extraction part is limited to simple removal. However, more sophisticated methods of information extraction are required to transform the documents' content into a standardized form. Furthermore, existing systems are evaluated using

prepared test data sets (cf. [119]) and thus face the problem that they were not designed to handle systematic errors such as optical character recognition (OCR) errors which are the results of the digitization of paper-based health records. Other issues include the correct recognition of misspelled words, unknown names and addresses, and words with multiple meanings.

To fulfill all three requirements, a combined approach of PHI identification and information extraction is required to produce de-personalized medical documents in a common data format. To improve the accuracy of the PHI identification process, it is best to combine individual techniques (cf. [50]) to circumvent their individual weaknesses.

4.2 Methodology

MEDSEC (MEDical records for SECondary use) is a system to create privacy-preserving health records for secondary use. It is designed to automatically convert large amounts of existing archived health records into pseudonymized HL7 CDA documents. In particular, it involves the following four phases (Figure 4.1):

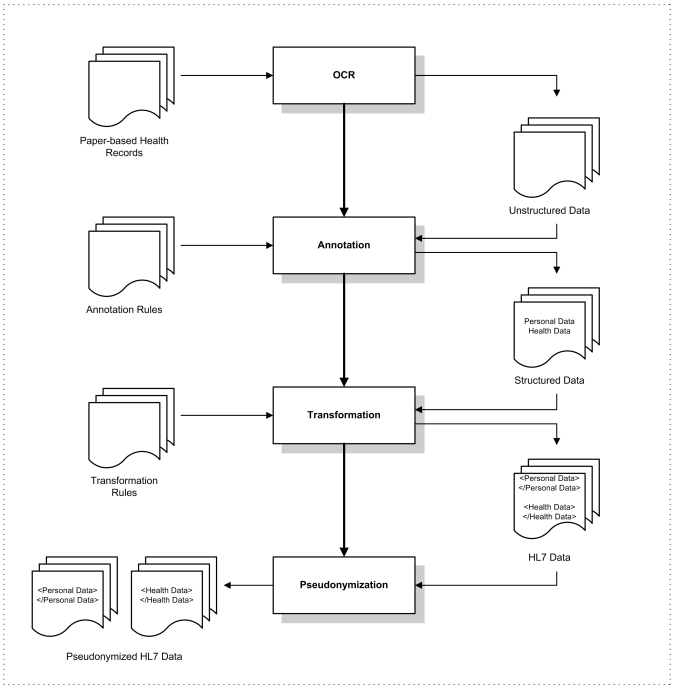


Figure 4.1: Phases of MEDSEC

OCR Since the majority of archived health records are still paper-based, they have to be made machine-readable. The OCR (optical character recognition) phase is tasked with converting the paper-based medical records (scanned images) into a machine-readable input string of characters. Furthermore, the output is extended with

structural annotations (paragraphs) that help in the annotation and transformation phases. The outcome is largely unstructured but machine-readable data.

Annotation In the annotation phase, the unstructured data is extended with annotations, identifying PHI elements using specific annotation rules and techniques which take both content and context of text elements into consideration. The annotation makes use of regular expressions, lists, and other domain knowledge to create structured data.

Transformation The structured data is further processed in the transformation phase and converted into standardized HL7 CDA documents. The transformation relies on the annotations added in the previous phase to identify the elements of interest using domain knowledge about structural compositions of the medical records.

Pseudonymization In the final phase, the HL7 CDA documents are split into personal and health sections and pseudonymized. Furthermore, any PHI elements identified in the annotation phase that are within the health section are also filtered out so that the pseudonymized health sections can be used for secondary use.

The system is designed to process documents with minimum interaction with a human operator once the system has been set up properly (e.g., conversion rules, input sources). International research projects such as EHR4CR[1] and EURECA[2] also investigate the utilization of existing electronic health records (EHR) for secondary use, though at a much larger scale. Although some aspects are similar (such as the application of natural language processing techniques and the consideration of privacy issues), these projects focus more on the interoperability of systems and the actual identification and extraction of useful medical information for research, whereas MEDSEC's goal is the total conversion of paper-based archives into a digital up-to-date document type (CDA) combined with de-identification and pseudonymization.

In the following, the individual logic modules responsible for the execution of their corresponding phases are described in detail.

4.2.1 OCR

The OCR module is responsible for converting paper-based health documents in the form of bit-mapped images (scans of the printed paper documents) into a continuous character stream. It is a support module which is only necessary when the documents are not available in machine-readable text form yet. Otherwise, the OCR phase can be completely skipped. As technical backbone, the module relies on the third-party OCR engine Tesseract[3] (Version 3.00), an open source OCR engine originally developed by Hewlett Packard and now published under the GNU Apache license 2.0. An overview of the OCR module is shown in Figure 4.2.

[1] www.ehr4cr.eu
[2] www.eurecaproject.eu
[3] https://code.google.com/p/tesseract-ocr/

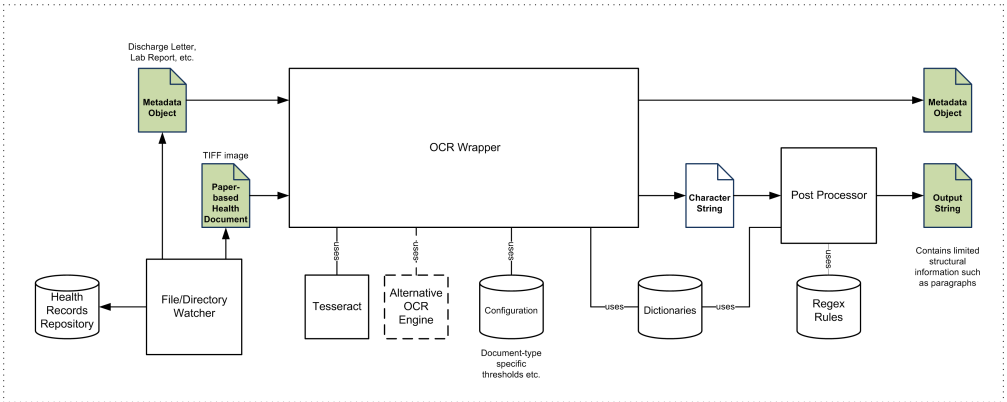


Figure 4.2: OCR module

A file/directory watcher monitors a specific folder for available medical records and fetches them to the main OCR wrapper. Since version 3.00, Tesseract includes a page layout analyzer which highly improves the information's 'cohesion' within a document. The page layout analyzer creates a pre-scan of the page to identify individual text blocks, processes them, and forwards them to the actual character recognition logic. Thereby, each text block can be processed individually, which allows grouping relevant information together (previous instances of Tesseract could only recognize text in a continuous stream). As the page layouts of different types of medical records vary substantially, document-type specific configurations need to be created which determine threshold values concerning distances (pixels) for detecting lines, blocks, or tables and others. These settings also define which text blocks are of particular interest and which could be ignored or need to be handled differently, such as header and footer lines. If, for example, header and footer lines are not collected and added to the output string at the end, they would interfere with page breaks. The thresholds also need to factor in skewness in the scanned images so that the text blocks are still accurately identified. These text blocks (e.g., paragraphs) are also included in the output strings as double line breaks (empty line) which are useful especially in the transformation phase.

Still, OCR engines are susceptible to misinterpretation of characters, especially when the scanned images are of less quality or are otherwise difficult to process (special backgrounds, etc.). In order to counter that, several error correction mechanisms are included. Comparison of the words with standards such as general purpose or domain-specific dictionaries improves accuracy. In addition, a post-processing step is added which looks for typical recognition failures such as telephone numbers or patient identification numbers or other identifiers which cannot be easily checked with the help of general purpose dictionaries. The post processor relies on specific regular expressions (regex) to identify typical characters (combinations) or well-known character mix-ups like the number 0 versus the capital letter O. Matching one of the regex rules results in the replacement of the characters or a re-evaluation of the word in question against dictionaries. These rules are usually required to be document-type specific. For example, if an internal

patient identifier is known to be composed of a capital letter followed by six digits and the document contains a section with a capital letter O within five digits, then it is highly likely that this is actually a misinterpreted number 0. If the capital letter O was at the beginning of the digit sequence, then it would rather be correct if followed by six digits.

4.2.2 Annotation

While there are no explicit European standards regarding the protection of PHI, the Safe Harbor method of the HIPAA Privacy Rule defines 18 PHI identifiers that have to be removed from the health record [80]: (i) names, (ii) locations, (iii) dates, (iv) ages greater than 89, (v) telephone numbers, (vi) fax numbers, (vii) email addresses, (viii) social security numbers, (ix) medical record numbers, (x) health plan beneficiary numbers, (xi) account numbers, (xii) certificate numbers, (xiii) vehicle identifiers, (xiv) device identification numbers, (xv) URLs, (xvi) IP addresses, (xvii) biometric identifiers, and (xviii) any other unique identifying number, code, or characteristic such as full face photos. Alternatively, the Expert Determination method involves a domain expert who decides on which elements are to be considered as PHI to adapt to the documents' domains and to increase data expressiveness and/or reduce the risk of reidentification. Still, the Safe Harbor list can act as a starting point for further refinements and thus is used as primary PHI list for MEDSEC.

The annotation module is mainly rule-based with the option of adding machine learning and recognizes PHI in the character strings produced by the OCR module, after the strings are tokenized, with one token representing a word or a single symbol. Then the following combination of recognition techniques are used:

Metadata (MD) If metadata such as patient name and medical record number is provided with the health record, tokens are matched against it to identify PHI.

Lookup tables (LT) Each token is matched against lookup tables. Each table is assigned to a specific PHI class (e.g., first name).

Pattern matching (PM) Rules, regular expressions, and general patterns are used to identify the PHI class (e.g., date identification).

Machine learning (ML) PHI is recognized based on a feature set and the corresponding training set.

Table 4.1 shows which techniques are used to recognize the PHI classes. The recognition techniques are not used in isolation. Instead, each technique uses the output of the previous techniques. For instance, pattern matching uses preliminary classification results of the lookup table matcher. Depending on the PHI class, each recognition technique has a different confidence rating indicating its overall rating of success for a correct annotation. For example, lookup table matching results would have a higher priority than certain pattern matching results at the recognition of names due to access to a statistical database including all first and last names of the current population of a country. In

PHI Class (tag)	MD	LT	PM	ML
First Name (firstName)	x	x	x	x
Last Name (lastName)	x	x	x	x
Age > 89 (age)	x	x	x	x
Street Name (street)		x	x	x
House Number (houseNo)			x	x
Postcode (zip)		x	x	x
City Name (city)		x	x	x
Organization (org)	x	x	x	x
Suborganization (subOrg)	x	x	x	x
Date (date)	x		x	x
Telephone Number (telNo)			x	x
Fax Number (faxNo)			x	x
Email Address (email)			x	x
Social Security Number (ssNo)	x		x	x
Medical Record Number (mrNo)	x		x	x
Health Plan Beneficiary No. (hpbNo)			x	x
Account Number (accountNo)			x	x
Certificate/License Number (licenseNo)			x	x
Vehicle Identifier (vehicleId)			x	x
Device Identifier (deviceId)			x	x
URL (url)			x	x
IP Address (ip)			x	x
Biometric Identifier (bioId)			x	x

Table 4.1: PHI class - recognition technique mapping

other cases, the exclusion of certain techniques would be beneficial to prevent too many false positives. The email address, for instance, is more suited for pattern matching (due to the @ symbol) than for lookup tables (high probability of being incorrectly identified as names).

Figure 4.3 provides an overview of the individual components of the annotation module: The initializer simply sets up the input string. The tokenizer then annotates tokens and whitespace tokens within the string. *Tokeniser.rules* defines how different kinds of tokens should be extracted from the string. Besides tokenization, the following basic token features are extracted: (i) length, (ii) orthography (e.g., upperInitial), (iii) kind (e.g., word or symbol), and (iv) the actual content of the token. Each token is matched against lookup tables that contain potential PHI. The metadata is structured data that is provided with the (largely) unstructured input data. For example, the documents taken from the document archive in the case study (cf. Section 4.3) are attached to (incomplete) metadata records used for internal querying operations containing standardized information including patient-related information (internal patient

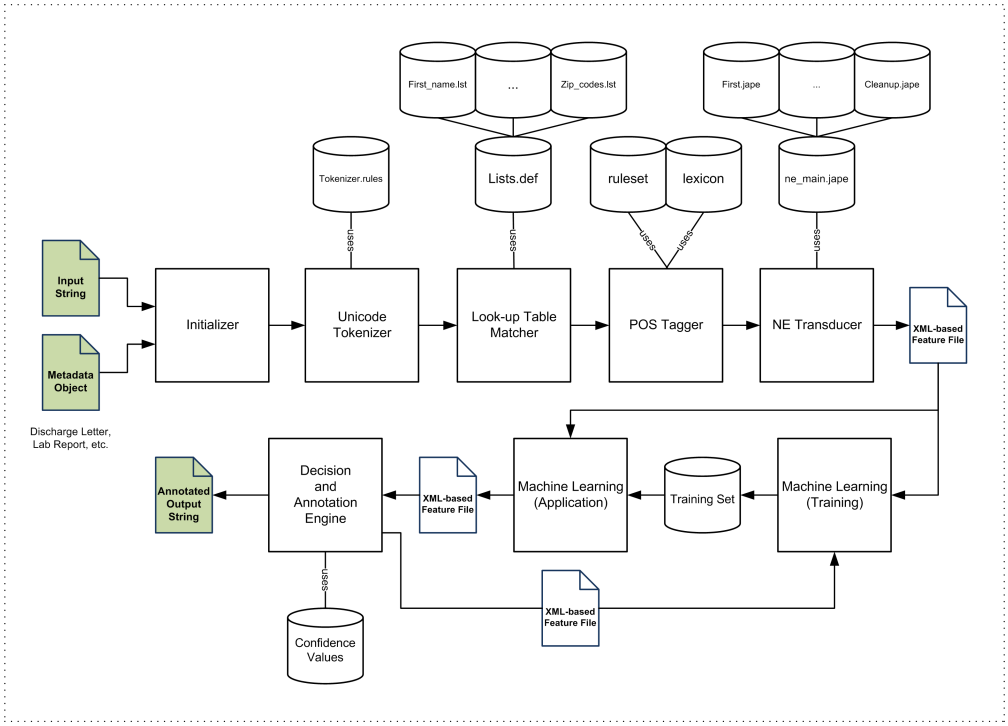


Figure 4.3: Annotation module

identifier, academic title, date of birth, etc.) and case-related information (date values of in-patient stays, medical record number, etc.). Each metadata field is mapped to a PHI class and matched against the tokens found in the unstructured data. In case of a positive match, the token is annotated according to the corresponding PHI class. The lookup tables include the following lists (compiled for Austrian health records):

First name A list of all first names registered in Austria between 1984 and 2006 has been provided by Statistics Austria (31690 entries).

Last name Based on an electronic Austrian phone book we compiled a list of last names (210125 entries).

Street name List provided by Statistics Austria (100318 entries).

Postcode List provided by Statistics Austria (1893 entries).

City name List provided by Statistics Austria (2794 entries).

Organization List of domain-specific organizations, i.e. hospitals (268 entries).

In general, multiple annotations are possible, e.g., one token can be annotated as organization and last name at the same time. In this case, this ambiguity can be solved in the next stages of the annotation process using the other annotation techniques.

The part-of-speech (POS) tagger annotates tokens with their part-of-speech (e.g., verb and noun) and their word stem. The module relies on the Tree Tagger implementation and German rule sets and lexicons. Besides basic features provided by the tokenizer, POS annotations are necessary for satisfactory machine learning results. The named entity (NE) transducer relies on rule sets to recognize PHI elements based on their grammar or syntax. For each PHI class, pattern matching rules are defined that recognize PHI elements based on their syntax and existing metadata and lookup table annotations. While PHI classes such as 'Email Address' and 'IP Address' can be easily recognized by applying regular expressions, more sophisticated rules are required to recognize most of the remaining PHI classes. For example, an Austrian postcode is represented by a four-digit number. Simply applying a regular expression would falsely recognize other four-digit numbers (such as telephone extension numbers) as postcodes. Therefore, rules have to consider previous annotations (e.g., each four-digit token is classified as a postcode if the following token has been recognized as a city name). Paper-based health records containing text in natural language usually include descriptive health-related elements such as the words 'anamnesis' or 'therapy'. These descriptors are also utilized by the rules. For example, it is very likely that the token preceded by the tokens 'First' 'Name' ':' represents a first name.

machine learning makes it possible to recognize unknown or misspelled terms within unstructured data. The annotation module uses the LibSVM (Library for Support Vector Machines) implementation and the GATE Tagger Framework to train and apply machine learning models for PHI recognition. The machine learning component uses the features produced by the previous modules as input features:

- Content of the token and its surrounding tokens, (-2/+2) denoting two preceding and two succeeding tokens, numbers determined by empirical tests
- Orthographic features (upper case, lower case, initial letter case) of the token and its surrounding tokens (-2/+2)
- Length of the token
- Part-of-speech of the token and its surrounding tokens (-2/+2)
- Kind (letter, numeric, symbol, etc.) of the token and its surrounding tokens (-2/+2)
- Lookup type of the token and its surrounding tokens (-2/+2)
- PHI class of the token and its surrounding tokens (-2/+2)

In addition to the input features of the actual document, the learner relies on training data to calculate annotation suggestions. Therefore, the final output of the decision and annotation engine is fed back to the training component of the learner to extend the training data set and to increase the accuracy of the learner.

This decision and annotation engine is responsible for the final annotation decision for each token and factors in the suggestions made in the previous stages of the process. All

PHI Class	Rule	Conf. Value
First Name	TitleNameRule	1.2
First Name	FirstBeforeLast	0.2
Street Name	AddressWithStr	1.2
Date	DatePointStartDay	1.2

Table 4.2: PHI class/rule confidence values (excerpt)

annotation suggestions are stored in an XML-encoded GATE document format. The NE transducer and its rules produce annotation suggestions with different quality depending on the PHI class. Therefore, confidence values are determined for each rule processed by the NE transducer. The confidence values are stored in a separate table and are linked to the PHI class (example shown in Table 4.2) by the rule identifier. These confidence values have been assigned based on multiple empirical experiments.

The final decision on the determination of the selected annotation candidate (SAC) is made by using the following formula to choose the candidate (C) with the highest aggregated recognition values:

$$SAC = C\left(max_x\left\{\sum_{i\in RT} x_i^{C_1}, ..., \sum_{i\in RT} x_i^{C_n}\right\}\right) \quad (4.1)$$
$$\text{with } x \in \{0,1\} \text{ for } i = LT \text{ and}$$
$$x \in \{0, 1.5\} \text{ for } i = MD \text{ and}$$
$$0 < x \leq 1 \text{ for } i \in \{PM, ML\}$$

where $RT = MD, LT, PM, ML$ denotes the list of PHI recognition techniques and $C_1, ..., C_n$ the individual annotation candidates.

This additive calculation scheme clearly favors PHI classes assigned to multiple recognition techniques. Emphasis is laid on the selection of the proper recognition techniques to benefit from high probabilities of correct annotation with certain technique/PHI class combinations (like with names and organizations and complete lookup tables). Using the results of several recognition techniques makes it possible to address the word sense disambiguation problem. While lookup table matcher and NE transducer focus on the local token context, the learner provides data on the global context of the token (i.e. in the context of the entire training data). For further processing, annotated health records are stored in a compact XML structure. PHI is annotated according to the tags described in Table 4.1 (e.g., a first name is annotated by the 'firstName' tag). The 'p' tag is used to represent paragraphs within the health record as shown in the following example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<document>
 <p>
  Anamnesis: Mrs. <lastName>Doe</lastName>
```

```
<firstName>Jane</firstName> went for a walk on the weekend of...
 </p>
</document>
```

4.2.3 Transformation

The transformation module primarily consists of the mapping processor (cf. Figure 4.4) which accepts the annotated input string produced in the previous phase and the metadata object. It relies on specific XML-encoded rules that identify and extract the relevant information from the input string and inserts the extracted content into HL7 CDA header and body templates which are then merged in order to create the CDA candidate document. The templates are organized into logical CDA sections (recordTarget, diagnosis, etc.) containing static text bodies and references where the extracted information should be inserted into. The CDA candidate document is then fed into the CDA validator which checks the candidate document for any missing elements and collects the notifications as feedback. The module's final output comprises two versions of the CDA document, one extended version as input for the pseudonymization including all annotations and another one with extracted annotations for archiving purposes.

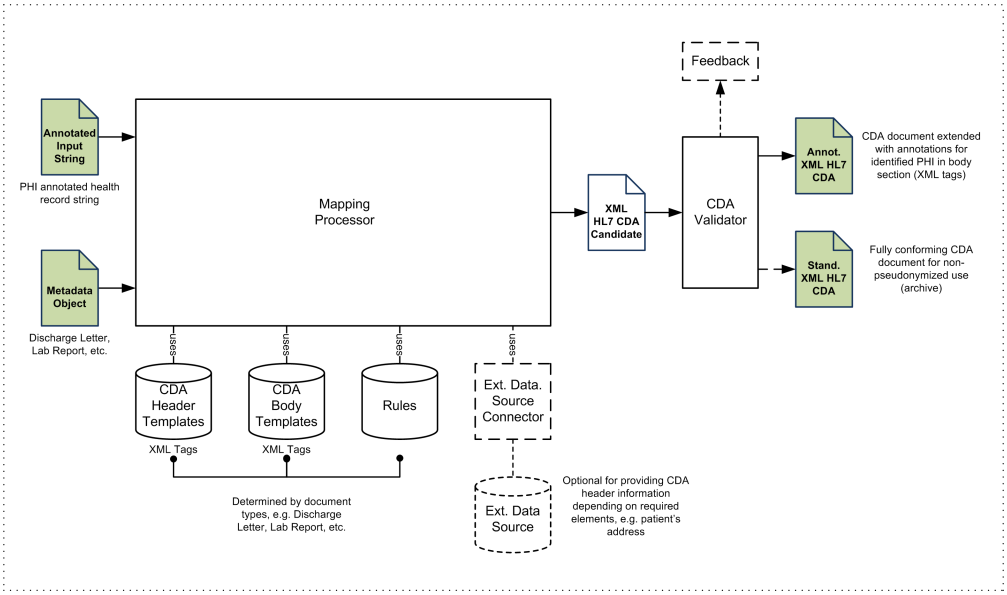


Figure 4.4: Transformation module

The underlying transformation methodology is completely rule-based to exploit the similarity of medical documents (e.g., discharge letters almost always contain sections including diagnosis, reason for visit, procedures, etc. separated into logical units like paragraphs), which makes it easier to write rules and, thus, to achieve results faster without a large training set. Separating transformation rules from target document templates also facilitates template reuse when adapting rules to a different document set

(e.g., different layouts of medical records depending on the creation date). Also, designing the transformation rules' syntax and semantics allows non-technical domain specialists to develop sophisticated rules covering all requirements without having to learn complex formalisms (e.g., XSLT combined with complex XPath expressions). The rules make use of both structural and content-related knowledge and require the input string to be (i) partitioned into semantically-contained text blocks such as paragraphs (section boundaries identification marked with <p> tags) and (ii) annotated with prespecified entity classes such as names or dates (named entity recognition). Each text block can have multiple named entities, again annotated with XML tags (e.g., <a_firstname>[4]). The document templates are organized into logical CDA sections (recordTarget, author, diagnosis, etc.) and contain the static text body (predefined) and references where the extracted information should be inserted into. Templates are selected and composed depending on the document types (discharge letter, lab results, etc.). Each of the templates is assigned one or more rules which are expressed in XML as well and contain subrules for each entity-of-interest in each template. Multiple rules for a single template account for different document subtypes (e.g., layout changes of discharge letter in the course of time).

Considering a typical discharge letter, the rules must be able to meet the following requirements (examples in parenthesis):

- Identification and extraction of single named entities (names, locations).
- Distinction between different instances of the same entity class (patient name vs. health professional name).
- Composition of entities (social security number with birth date).
- Limited sections of text blocks (ICD codes).
- Complete text blocks (diagnosis).
- Multiple successive text blocks (medical history extending over multiple paragraphs).

The transformation process is sketched in Figure 4.5:

1. Identification of the document (sub)type and selection of the appropriate rules and templates.
2. For each template:
    a) Identification of the region-of-interest (one or more paragraphs) using region structure (combination of annotations) and/or content (regular expression) information as decision tools.
    b) Extraction of the actual content-of-interest from the region-of-interest using either region structure or content information as filters.

[4] 'a' for annotation to distinguish these tags from any preexisting tags in the source input string

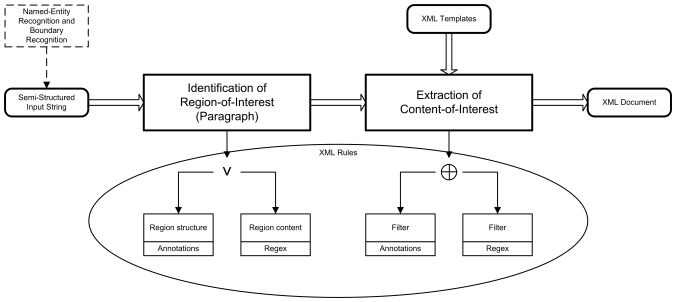


Figure 4.5: Transformation process

c) Insertion of the content-of-interest into the corresponding sections in the template.

d) Continuation with the next template until all templates and corresponding rules are processed.

3. Composition of the templates into a full CDA document.

In the following, templates and rules, their semantics, and text processing effects are described in detail. As the syntax is XML-based, both templates and rules must conform to XML schema definitions. For better overview, the template and rule structures are represented as figures with boxes as XML elements (nodes), where element attributes are shown below and element text content on the right. Parent/child relationships of the nodes are expressed by connecting lines with cardinality indicators, extended with sequence/choice bars if applicable.

Templates

A template represents a building block (e.g., diagnostic section) that needs to be filled with the extracted information and combined with other templates to form the complete CDA document. Each <template> (see Figure 4.6) has a unique 'ID' and contains the <itemList> and <content> sections. While <content> contains the static text building blocks with empty sections that need to be filled with information extracted from the input string, <itemList> contains a set of <itemPath> elements corresponding to each empty field in the <content> section. An <itemPath> definition has an 'ID', optional 'prefix' and 'suffix' attributes (i.e. static text that is added to the extracted information like area codes), and an 'opt' indicator expressing whether the particular field is optional (e.g., patient names are mandatory while telecom values are not). The XPath expression specifies the field's location in the <content> section.

Rules

A rule encodes the information of how to identify (region-of-interest) and extract the relevant entities (content-of-interest) from the source input string. Each <rule> (see

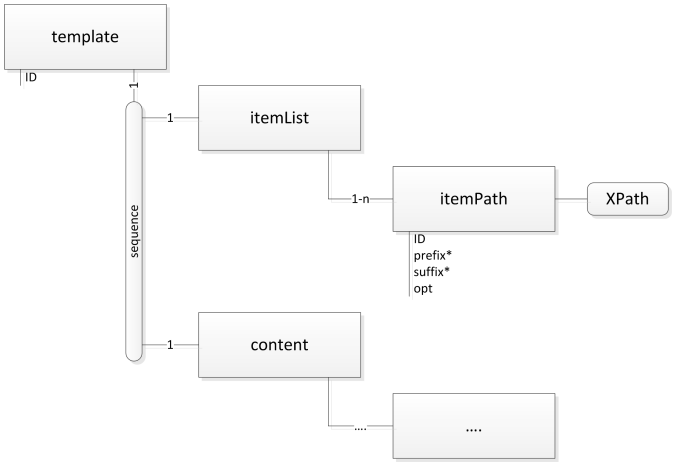


Figure 4.6: Template structure

Figure 4.7) must correspond to a particular template indicated by the 'targetTemplate' attribute. The other attribute defines a document subtype for which the rule is applicable (e.g., discharge letters from different wards within a hospital). Each rule contains one or more <region> nodes representing the regions-of-interest (i.e. paragraphs) where the <conditions> section denotes the structural and content-related conditions for finding the correct region and <contentMapping> the actual content that needs to be extracted from the region(s) and inserted into the corresponding parts of the CDA templates.

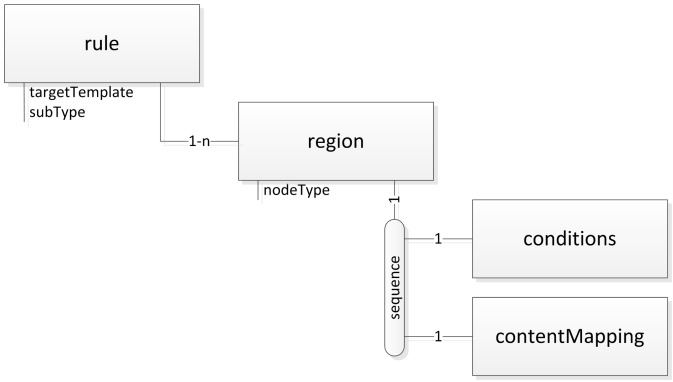


Figure 4.7: Rule structure

The region's 'nodeType' attribute defines different ways of how to find the particular nodes, i.e. paragraphs in the input string, of the regions-of-interest. Depending on 'nodeType', a different set of child <(begin/end)nodeCondition> elements is necessary in the <conditions> section (see Figure 4.8). Types, required children in the <conditions> section, and purposes are as follows:

- The 'single' type states that the region consists of a single paragraph only and, thus, requires a single <nodeCondition> element only. This type is useful when the node's composition is known relatively well.
- The 'singleSelected' type states that the region consists of multiple paragraphs that do not necessarily occur in immediate succession in the source input string. Only a single node per node condition is returned. It requires two or more <nodeCondition> elements and is used to get multiple nodes where each node condition refers to a single node and the nodes' compositions are known relatively well.
- The 'multiSelected' type states that the region consists of all nodes matching any node condition and requires one or more <nodeCondition> elements. It is used to get multiple nodes which may contain any known regular expression keyword and/or tag (see below).
- The 'multiContinuous' type states that the region contains all nodes between a beginning node, <beginNodeCondition>, and an ending node, <endNodeCondition>. The 'conditionType' attribute determines whether the begin/end nodes are included in the region or not. The 'multiContinuous' type is used to get all nodes within two known boundaries.

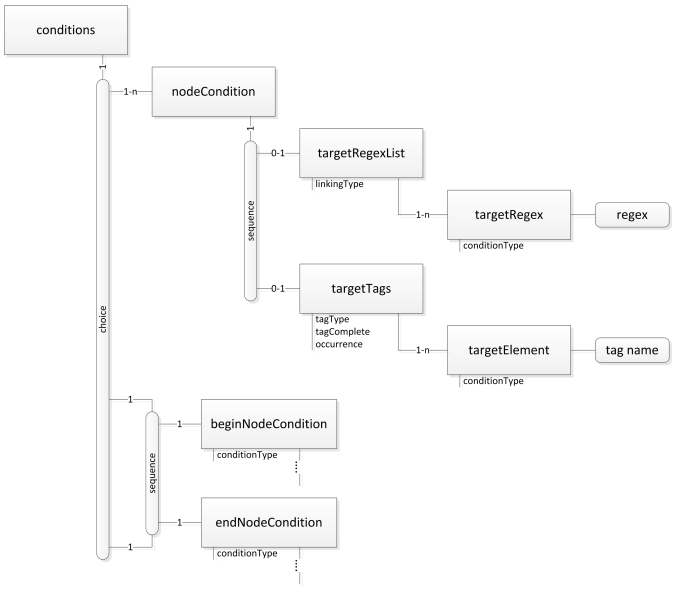


Figure 4.8: Rule - conditions structure

The <nodeCondition> element contains a <targetRegexList> and/or a <targetTags> element. The former describes a condition on the paragraph's content, while the latter states the set of required XML tags, i.e. annotations within the paragraph. The <targetRegexList> contains one or more <targetRegex> elements with the actual regular

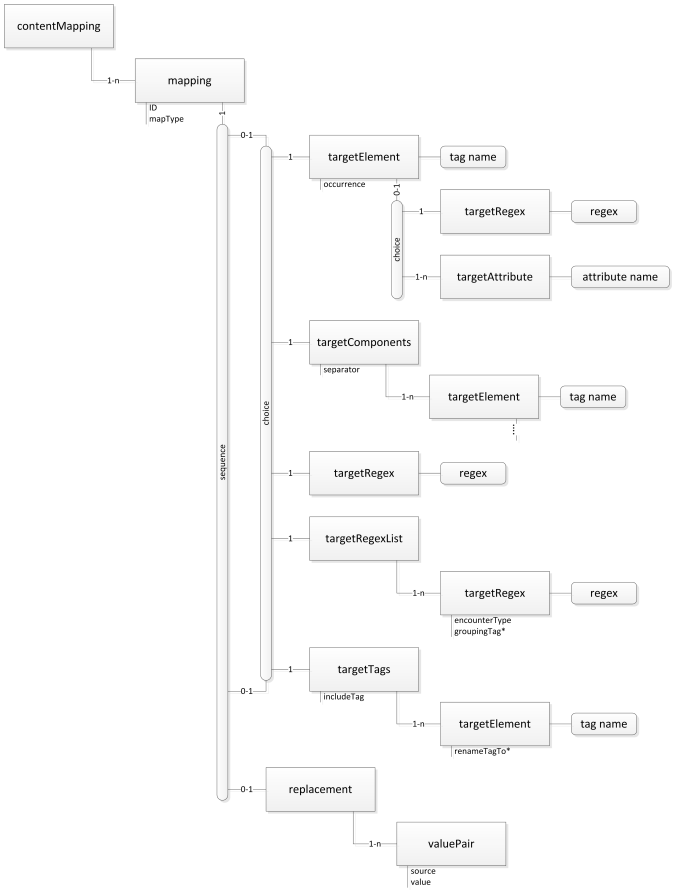


Figure 4.9: Rule - content mapping structure

expression as text value. The attributes 'linkingType' and 'conditionType' indicate how the individual regular expressions are linked (and/or) and whether the particular regular expression must or must not yield a match in the paragraph's text in order to be part of the region-of-interest (include/exclude). Similarly, <targetTags> contains one or more annotations defined as <targetElement> and tag name as text values which have to occur in either a particular sequence or in any sequence ('tagType'). The 'tagComplete' attribute defines whether the paragraph must not contain any annotation other than that of <targetElement>, while 'occurrence' states the desired occurrence of the annotation combination (e.g., 'occurrence' = '2' matches the second paragraph with the particular annotation combination).

Apart from the 'conditionType' as attributes, <beginNodeCondition> and <endNode-Condition> are composed just like the <nodeCondition> elements.

While <conditions> indicate the conditions under which the paragraphs belong to the region-of-interest of the current rule, <contentMapping> defines which information to actually extract from the region-of-interest. The section contains one or more <mapping>

elements (Figure 4.9). The attribute 'ID' indicates where to insert the extracted information into the templates (matching the <itemPath> 'ID' attribute), while 'mapType' defines how to extract the information, defined as follows:

- The 'singleElement' type represents a 1:1 mapping of an annotation tag's content (determined by tag name and its 'occurrence' within the region-of-interest, e.g., second occurrence of <a_firstName> as the patient's first name) to a particular section within the CDA template. The content can optionally be filtered by a regular expression or composed of the annotation tag's attributes instead of the text content (indicated by one or more <targetAttribute> elements).

- The 'composedElement' type represents a content written into the CDA template composed of multiple annotated <targetElement> tags, separated by a 'separator' (default is an empty string), e.g., the composition of the Austrian ten-digit social security number using the four-digit version combined with the person's birth date. The <targetElement> elements can again be filtered by regular expressions or attribute values.

- The 'fullContent' type simply refers to extracting the whole content of the region-of-interest into the corresponding section of the CDA template. It can (optionally) be filtered by a single <targetRegex>. An example of 'fullContent' is to copy the complete narrative paragraph of a diagnosis to the CDA template, while it may be filtered to extract the ICD codes only.

- The 'multiContent' type refers to multiple elements within the region-of-interest that are inserted into the CDA template separated by a default grouping tag[5] unless specified otherwise. The content can be filtered by either a <targetRegexList> or <targetTags>. The <targetRegexList> contains a set of one or more <targetRegex> elements with attributes 'encounterType' and 'groupingTag'. The former attribute distinguishes between extracting only the first regular expression-matching string or all matching strings within the region-of-interest, while the optional latter attribute overrides the default grouping tag with an alternative tag for each individual regular expression. <targetTags> contains the set of <targetElement> tags whose contents are to be extracted. The 'includeTag' attribute indicates whether the elements' original tags should be copied to the template too (in this case, the default grouping tag is replaced with the original one). The optional attribute 'renameTagTo' allows to individually rename the tag to an arbitrary tag for each <targetElement>. Both <targetTags> and <targetRegexList> have their fields of uses, e.g., to extract ICD codes from a diagnostic text spanning over multiple paragraphs, depending on how these are annotated by the annotation engine: If the tags are already annotated, <targetTags> can be used to extract them with potentially renaming the annotation tags with arbitrary tags matching the CDA standard. If the annotation is not able to annotate the tags, <targetRegexList> may contain the regular expressions to

[5] As this is only used in CDA body templates, we simply use <paragraph> here.

extract them (in this case with 'encounterType' = 'all'), again with an optional renamed grouping tag.

All mapping sections can have an additional and optional <replacement> section containing <valuePair> elements. These elements refer to text elements ('source') that should be replaced with another text ('value'), e.g., if the word 'female' is encountered within a <gender> tag, it should be replaced with 'F' before being inserted into the patient template of the CDA header section.

Algorithm

The core of the transformation process is the algorithm to identify the regions-of-interest, followed by the extraction of the content-of-interest. For the sake of clarity, we focus here on the condition and mapping types. The overall procedure is depicted in Algorithm 4.1.

Algorithm 4.1: Overall transformation algorithm

```
   Input: Annotated source input string
   Output: Converted CDA document
 1 Identify document type and date;
 2 Select and load the corresponding templates;
 3 Select and load the corresponding rules;
 4 forall the Rules do
 5 |   Identify the corresponding template;
 6 |   forall the Regions in the current rule do
 7 |   |   Apply Conditions;
 8 |   |   Apply Content mappings;
 9 |   end
10 end
11 Compose the filled templates to a CDA document according to the document type's
   specifications;
12 Validate CDA document:
13     forall the 'itemPaths' in all filled templates do
14 |       if Corresponding section in template is empty AND mandatory then
15 |   |       Add to validation report;
16 |       end
17     end
```

The process consists of identifying and loading the correct rules and templates (lines 1-3), iterating all rules and applying the conditions and content mappings (lines 4-10), composing the filled templates to a full CDA document and validating the document for missing sections (lines 11-17).

Algorithm 4.2: Apply conditions algorithm

```
   Input: Region
   Input: Regions-of-interest
 1 if 'nodeType' = 'single' then
 2 |   foreach Paragraph do
 3 |   |   if Current paragraph matches the tag and/or regex condition then
 4 |   |   |   Mark it as region-of-interest;
 5 |   |   |   Stop;
 6 |   |   end
 7 |   end
 8 end
 9 else if 'nodeType' = 'singleSelected' then
10 |   foreach Paragraph and node condition do
11 |   |   if Current paragraph matches the CURRENT tag and/or regex condition then
12 |   |   |   Mark it as region-of-interest;
13 |   |   |   Continue with next paragraph AND next node condition;
14 |   |   end
15 |   end
16 end
17 else if 'nodeType' = 'multiSelected' then
18 |   foreach Paragraph do
19 |   |   if Current paragraph matches ANY tag and/or regex condition then
20 |   |   |   Mark as region-of-interest;
21 |   |   |   Continue with next paragraph;
22 |   |   end
23 |   end
24 end
25 else if 'nodeType' = 'multiContinuous' then
26 |   foreach Paragraph do
27 |   |   if Current paragraph matches the 'beginNode' tag and/or regex condition then
28 |   |   |   Mark as begin node;
29 |   |   |   Continue with next paragraph and skip checking for 'beginNode' conditions
   |   |   |   beginning with the next paragraph;
30 |   |   end
31 |   |   if Current paragraph matches the 'endNode' tag and/or regex condition then
32 |   |   |   Mark as end node;
33 |   |   |   Stop;
34 |   |   end
35 |   end
36 |   Mark all paragraphs between the begin and end node as regions-of-interest;
37 |   Include the actual begin and end node depending on the 'conditionType';
38 end
```

A more detailed view of the 'Apply Conditions' operation is depicted in Algorithm 4.2. It distinguishes between different node types (cf. Section 4.2.3): 'single' looks for the first paragraph (i.e. section in the source document enclosed in <p> tags) which matches the tag and/or regex condition (lines 1-8); 'singleSelected' returns all matching

paragraphs per each node tag/regex condition (lines 9-16); 'multiSelected' in turn yields all paragraphs that match any tag/regex condition within the current rule (lines 17-24), while 'multiContinous' first searches for the begin node, then for the end node (in the remaining paragraphs) by their corresponding tag/regex conditions and marks all paragraphs between them (optionally including them) as region-of-interest (lines 25-38).

Algorithm 4.3: Apply content mappings algorithm

```
   Input: Regions-of-interest
   Output: Filled templates
 1 foreach Content mapping do
 2     if 'mapType' = 'single' then
 3         Extract content-of-interest by tag and filter by regex (if applicable);
 4         Insert the content into the corresponding section of the template identified by the
           'itemPath';
 5     end
 6     else if 'mapType' = 'composed' then
 7         Extract all contents-of-interest by tag and filter by regex (if applicable);
 8         Compose the parts with the given separator (if applicable);
 9         Insert the content into the corresponding section of the template identified by the
           'itemPath';
10     end
11     else if 'mapType' = 'fullContent' then
12         Extract the whole content of the region-of-interest and filter by regex (if
           applicable);
13         Insert the content into the corresponding section of the template identified by the
           'itemPath';
14     end
15     else if 'mapType' = 'multiContent' then
16         Extract the whole content of all regions-of-interest and filter by either tags or regex
           (if applicable);
17         Group the content with the given 'groupingTag' OR use/replace the original tag
           depending on 'includeTag' and 'renameTagTo';
18         Insert the content into the corresponding section of the template identified by the
           'itemPath';
19     end
20 end
```

Algorithm 4.3 shows how the content mappings are applied. Depending on the 'mapType' the content-of-interest is extracted differently from the region-of-interest: 'single' simply extracts the tag content (possibly filtered by regex, lines 1-5), while 'composed' combines multiple tag contents to a single element in the order defined by the mapping conditions' order (lines 6-10); 'fullContent' in turn simply selects the whole content of the region-of-interest (possibly filtered) as content-of-interest (lines 11-14), and finally 'multiContent' allows combining multiple sections in the region-of-interest, possibly filtered by either tags or regex, with predefined grouping tags to form a single CDA section entry.

Example

In the following example, a patient's name and address are extracted from a digitized and annotated medical discharge letter (source input string). The input string is formatted as XML document with paragraphs represented as enclosing <p> tags. Furthermore, the source document is also extended with prespecified tags including <a_title> for academic title, <a_firstName> for first names, <a_lastName> for last names, <a_date> for dates, and so on, by the annotation module (cf. Section 4.2.2). An excerpt of the source document is shown in Figure 4.10. Note that the annotation module only adds annotations while leaving the document structure as it is (line breaks, spaces, etc.).

```
<document>
...
<p>
To <a_title>Dr.</a_title> <a_firstName>Jane</a_firstName> <a_lastName>Roe</a_lastName>
<a_street>A Street</a_street> <a_houseNo>1</a_houseNo>
<a_city>A City</a_city>, A State <a_zip>12345</a_zip>
<a_country>A Country</a_country>
</p>
...
<p>
Patient: <a_firstName>John</a_firstName> <a_lastName>Doe</a_lastName>, born on <a_date day="01" month="01" year="1970">1970-01-01</a_date>
</p>
<p>
Home Address:
<a_street>Another Street</a_street> <a_houseNo>10</a_houseNo>
<a_city>A City</a_city>, A State <a_zip>12345</a_zip>
<a_country>A Country</a_country>
</p>
<p>
Diagnosis:
</p>
...
</document>
```

Figure 4.10: Example source document (excerpt)

The target template, i.e. the 'recordTarget' header section of the CDA document, is shown in Figure 4.11 and consists of the static segment specified by the CDA document standard with empty attribute/element sections (lower part) and the corresponding <itemPath> elements with the identifiers and XPath expressions (upper section). For example, the item 'nameGiven' has to be inserted into the empty element '/recordTarget/patientRole/patient/name/given'.

The rule, or more specifically, the particular region that is responsible for the patient's name and address is depicted in Figure 4.12: The <region> is of 'multiContinuous' type where the <conditions> section contains the specifications for the nodes beginning and ending the region-of-interest. For the beginning node, the particular paragraph has to include either the text segment 'born on' or 'date of birth'; it also contains both tags <a_lastName> and <a_date>. The attribute 'tagComplete' = 'false' denotes that other tags are allowed within this paragraph. Returning to the source document (Figure 4.10), the second paragraph matches this condition, as the first misses out the two regex and does not include the <a_date> tag either. Due to their common occurrence in medical documents, the use of keywords such as 'born on' in combination with tags is useful in solving the name disambiguity problem (e.g., patient's name vs physician's name) and distinguishing otherwise similar sections. The second condition in the rule for the

```
<template ID="recordTarget">
    <itemList>
        <itemPath ID="patID">/recordTarget/patientRole/id[1]/@extension</itemPath>
        <itemPath ID="patAuthority">/recordTarget/patientRole/id[1]/@assigningAuthorityName</itemPath>
        <itemPath ID="ssn10">/recordTarget/patientRole/id[2]/@extension</itemPath>
        <itemPath ID="ssnAuthority">/recordTarget/patientRole/id[2]/@assigningAuthorityName</itemPath>
        <itemPath ID="street">/recordTarget/patientRole/patient/addr/streetName</itemPath>
        <itemPath ID="houseNumber">/recordTarget/patientRole/patient/addr/houseNumber</itemPath>
        <itemPath ID="postal">/recordTarget/patientRole/patient/addr/postalCode</itemPath>
        <itemPath ID="city">/recordTarget/patientRole/patient/addr/city</itemPath>
        <itemPath ID="country">/recordTarget/patientRole/patient/addr/country</itemPath>
        <itemPath ID="phone" prefix="tel:" opt="true">/recordTarget/patientRole/patient/telecom[1]/@value</itemPath>
        <itemPath ID="mail" prefix="mailto:" opt="true">/recordTarget/patientRole/patient/telecom[2]/@value</itemPath>
        <itemPath ID="namePrefix" opt="true">/recordTarget/patientRole/patient/name/prefix/@qualifier</itemPath>
        <itemPath ID="nameGiven">/recordTarget/patientRole/patient/name/given</itemPath>
        <itemPath ID="nameFamily">/recordTarget/patientRole/patient/name/family</itemPath>
        <itemPath ID="sex">/recordTarget/patientRole/patient/administrativeGenderCode/@code</itemPath>
        <itemPath ID="birthDate">/recordTarget/patientRole/patient/birthTime/@value</itemPath>
    </itemList>
    <content>
        <recordTarget>
            <patientRole>
                <id root="2.16.840.1.113883.19.5.3" extension="" assigningAuthorityName=""/>
                <id root="2.16.840.1.113883.19.5.4" extension="" assigningAuthorityName=""/>
                <patient>
                    <addr use="HP">
                        <streetName></streetName>
                        <houseNumber></houseNumber>
                        <postalCode></postalCode>
                        <city></city>
                        <country></country>
                    </addr>
                    <telecom value="" use="HP"/>
                    <telecom value=""/>
                    <name>
                        <prefix qualifier="AC"></prefix>
                        <given></given>
                        <family></family>
                    </name>
                    <administrativeGenderCode code="" codeSystem="2.16.840.1.113883.5.1" codeSystemName="HL7 Administrative Gender"/>
                    <birthTime value=""/>
                </patient>
            </patientRole>
        </recordTarget>
    </content>
</template>
```

Figure 4.11: CDA header section: patientRole

ending node consists of the single regex keyword 'Diagnosis:' which matches the fourth paragraph in the source document. Because of the 'conditionType = 'exclude', this fourth paragraph needs to be excluded, thus the region-of-interest comprises the second and third paragraphs.

The actual content mapping involves just a simple extraction of tag contents ('mapType' = 'singleElement') for the first/last names and address components without any regex filtering and insertion into the template.

4.2.4 Pseudonymization

In the final phase, the (annotated) HL7 CDA document produced by the transformation module is pseudonymized to prevent reidentification. The pseudonymization method is based on the CDA scenario of PERiMETER (cf. Section 3.3.2), but without the authorization mechanism due to MEDSEC's focus on secondary use only. The research data manager de-pseudonymizes individual records as data owner when deemed necessary.

As shown in Figure 4.13, the actual pseudonymization is preceded by document fragmentation where the CDA document is split into multiple individual parts. Furthermore, the document fragmenter screens the CDA body section for the PHI annotations, extracts

```
<region nodeType="multiContinuous">
    <conditions>
        <beginNodeCondition conditionType="include">
            <targetRegexList linkingType="or">
                <targetRegex>born on</targetRegex>
                <targetRegex>date of birth</targetRegex>
            </targetRegexList>
            <targetTags tagType="all" tagComplete="false">
                <targetElement>a_lastName</targetElement>
                <targetElement>a_date</targetElement>
            </targetTags>
        </beginNodeCondition>
        <endNodeCondition conditionType="exclude">
            <targetRegexList linkingType="and">
                <targetRegex>Diagnosis:</targetRegex>
            </targetRegexList>
        </endNodeCondition>
    </conditions>
    <contentMapping>
        <mapping ID="nameGiven" mapType="singleElement">
            <targetElement>a_firstName</targetElement>
        </mapping>
        <mapping ID="nameFamily" mapType="singleElement">
            <targetElement>a_lastName</targetElement>
        </mapping>
        <mapping ID="street" mapType="singleElement">
            <targetElement>a_street</targetElement>
        </mapping>
        <mapping ID="houseNumber" mapType="singleElement">
            <targetElement>a_houseNo</targetElement>
        </mapping>
        <mapping ID="postal" mapType="singleElement">
            <targetElement>a_zip</targetElement>
        </mapping>
        <mapping ID="city" mapType="singleElement">
            <targetElement>a_city</targetElement>
        </mapping>
    </contentMapping>
</region>
```

Figure 4.12: A region section of a rule matching the 'recordTarget' template

them, and replaces them with general purpose tags. The extracted PHI elements are then collected in a separate body PHI XML document. In the figure, document fragmentation is shown on its highest level, (i) CDA header fragment, (ii) CDA body fragment, and (iii) extracted body PHI elements. However, the fragmentation level can be adapted if required (e.g., further fragmenting the body into individual CDA sections). After the actual pseudonymization operation by the pseudonymization engine, the resulting documents are stored in different storages: The PHI storage contains the CDA header and PHI documents, whereas the pseudonymized CDA body documents are stored in the PSY (pseudonym) storage. The PSY metadata contains the pseudonymization linkage information. In order to securely generate and encrypt the pseudonyms, a crypto service provider is used. There are different possibilities for implementing the crypto service provider such as in a software module with a file-based key store or in a fully-fledged hardware security module with tamper-resistant hardware and a secure key storage within the confinements of the hardware (cf. Section 3.2.3). The decision of how to implement the crypto service provider depends on the overall system's technical and organizational environment and the required security level.

As described in Section 3.2.1, the pseudonymization involves (randomly selected)

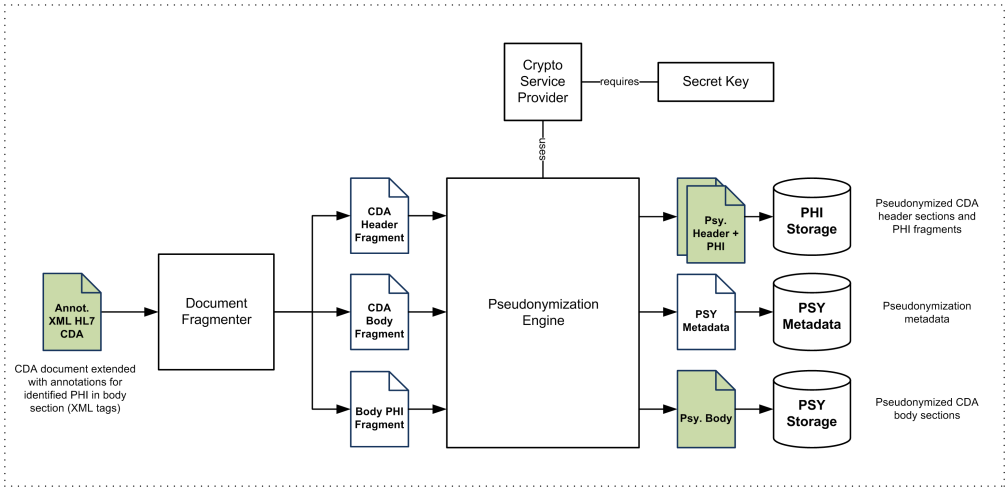


Figure 4.13: Pseudonymization module

pseudonyms for each fragment (in this case the PSN_{HE}, PSN_{BO}, and PSN_{PHI}) attached to the fragments (represented by their record identifiers RID_{HE}, RID_{BO}, and RID_{PHI}). While the mappings to the records are stored in cleartext, the link between the pseudonyms representing the links to reidentify the necessary document fragments to recover the complete CDA document is protected by encryption (structure shown in Figure 4.14). Assuming that the crypto service provider and the secret key are both protected, pseudonymized documents cannot be reidentified without explicit access to the secret key managed by the crypto service provider. Thus, while the PSY storage is intended to be fully accessible by any secondary user and the PHI storage and PSY metadata should be kept unavailable for them, the documents are still protected from reidentification in case of unwanted full disclosure of the storages.

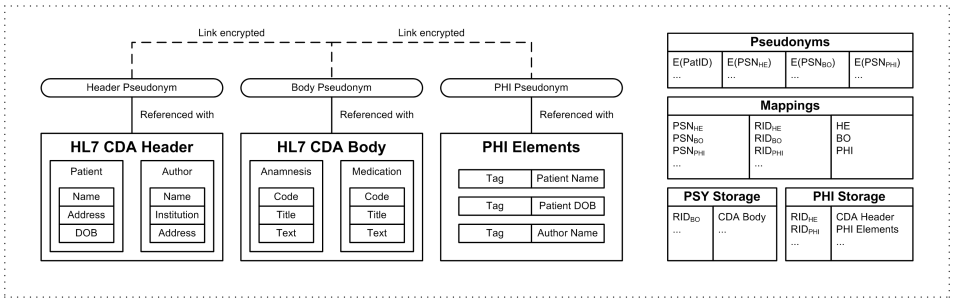


Figure 4.14: CDA/pseudonym structure

To de-pseudonymize a particular CDA body, the mappings table stores additional pseudonym types (HE, BO, PHI) that help to reidentify the correct associated header and PHI pseudonyms. Retrieving all documents of a particular patient can be achieved

by executing an exact match query over the encrypted patient identifier $PatID$. Thus, the logic also supports partial de-pseudonymization, i.e. retrieving multiple documents of the same patient without disclosing the patient's identity in the form of the CDA header and PHI elements fragment.

4.3 Implementation and evaluation

All four engines were implemented in Java as independent software modules executed by the XiTrust Business Server (XBS[6]), an advanced Java-based workflow engine which allows control flow changes and dynamic loading of software components on demand. The XBS also provided additional interfaces like the connector to the document archive and to the hospital information system necessary for the case study. The crypto service module provided by the XBS relies on standardized external interfaces which allow a software-based or a hardware security module as cryptographic backend. In our test case, we used a software implementation with a separate authentication module and key storage.

The case study involves the automated processing and conversion of paper-based discharge letters taken from an archive of an Austrian hospital. The discharge letters stored as TIFF images of the scanned pages included incomplete metadata (patient name, case number, etc.). The documents were taken from two different wards (internal medicine and surgery) of the same hospital. While the same configuration settings, lookup tables, and rules for the OCR and annotation modules were used, we created two sets of transformation rules for each ward in order to adapt to the different document layouts. The transformation rules required some localized changes such as the positioning and sequence of names. For example, in the first ward, the patient names were written as first names followed by the last names, while in the second ward, the order was the other way round. Another example involved the sequence of the elements in the letter head. In general, both wards included the same amount of information in their discharge letters, though there were minor structural changes that had to be taken into consideration. As the rules can be expressed in different levels of granularity (e.g., single rule for patient's first name, single rule for the complete diagnosis section of the discharge letter), fine-grained rules were created to account for these minor structural differences, while it was possible to reuse the remaining rules such as the one for the diagnosis (which always followed the salutation section of the letter).

In a production environment, the setup of the system would require some manual work before the automated conversion can commence. At first, a domain expert has to analyze the paper-based documents that have to be converted. Usually, the documents can be categorized into different classes (lab results, discharge letters, etc.). Apart from that, documents may also differ depending on their source of origin such as the hospital wards (see above). Once the document's structure and content have been identified, the annotation and transformation rules have to be devised. Since the content of a particular

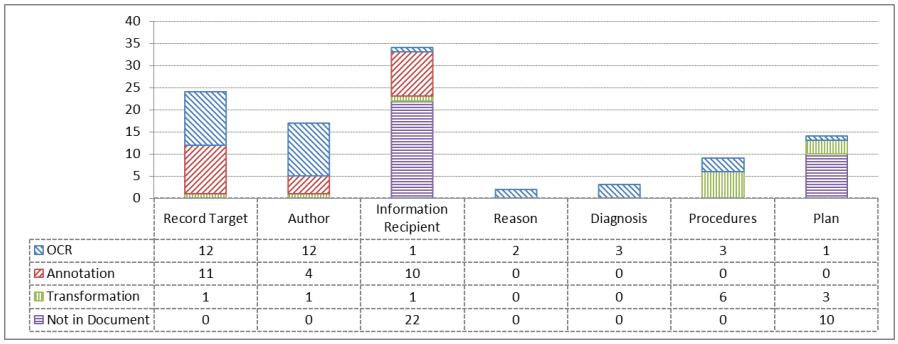
[6] www.xitrust.com

document class usually contains the same type of information (also specified by law), many annotation and transformation rule sections can be reused. Knowledge of the font types used as well as the layout of the scanned documents (such as the positioning of header and footer sections) can also help to fine-tune the OCR module. To improve the accuracy of the conversion process, the outcome needs to be analyzed manually for errors in order to improve the rules. This conversion-analysis-revision cycle needs to be done multiple times to reach a satisfactory rule set.

In our case study, we have applied following testing methodology: At first, we randomly selected 100 documents acting as the test corpus and placed them into a specific folder in order to trigger the OCR module's file watcher. The documents then underwent all four phases of the MEDSEC system, where the documents were collected after each phase. In parallel, we manually screened the OCR output and corrected any errors, annotated the documents manually and transformed the strings into CDA documents to create the gold standard to compare it with the output of the (unsupervised) automated MEDSEC test run. To test the overall robustness of the overall framework, we only used the paper-based documents and the metadata (if existing) as information sources; no external data sources like organizational data from hospital information systems were included. We also deactivated the machine learning components of the annotation module due to the limitations of available training data.

As the pseudonymization phase's outcome is deterministic and the application of symmetric cryptography (under the assumption of a securely kept secret key) is proven secure, we focused on the outcome's quality in terms of annotation and transformation. We split the CDA document into the following segments for examination: The header section's focus of interest included recordTarget (patient's personal information), author (usually the treating physician), and informationRecipient (physician receiving the discharge summary), while the body section included reason for visit (incidents, problems), diagnosis (results and outcome), procedures (medications and treatments), and plan of care (recommended medications). The results in Figure 4.15 show the overall erroneous sections due to missing elements or incorrectly classified entities. We identified a significantly higher percentage of total errors in the CDA header blocks (24%, 17%, 34%) than in the CDA body blocks (2%, 3%, 9%, 14%), though the errors in the header blocks were usually due to a single missing element such as a street name in an address block. These numbers were acquired by simply counting all errors including missing or incorrectly assigned entities.

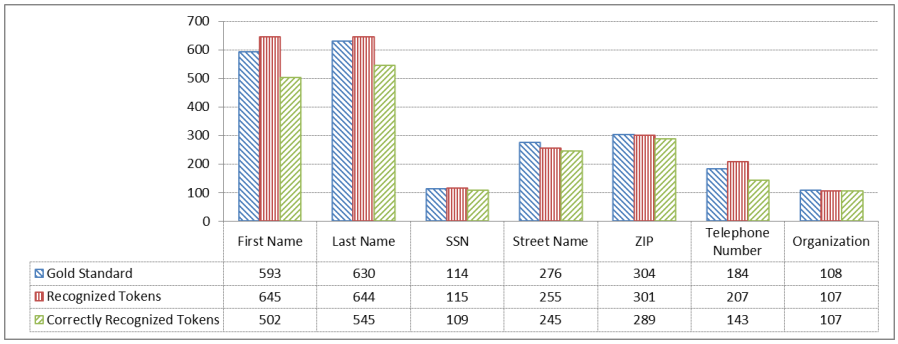
Besides, we analyzed each error and identified the corresponding cause: The majority of the errors was traced back to incorrect annotations (recordTarget and informationRecipient), partly due to recognition errors in the OCR module (recordTarget and author). Many errors in the recordTarget and author sections were due to incorrect (or missing) name annotations. An investigation of the original paper-based scanned images revealed the problem: First and last name annotation errors were the result of character recognition difficulties due to special greyed and dotted background boxes surrounding the patient names in the document. Another source of recognition failures was hand-written signatures placed directly over the printed names, which was the main cause for misidentified



	Record Target	Author	Information Recipient	Reason	Diagnosis	Procedures	Plan
OCR	12	12	1	2	3	3	1
Annotation	11	4	10	0	0	0	0
Transformation	1	1	1	0	0	6	3
Not in Document	0	0	22	0	0	0	10

Figure 4.15: Erroneous CDA sections

physician names. The transformation rules for the CDA header section mainly consisted of structural conditions (i.e. annotation combinations); missing or incorrectly annotated names therefore reduced the number of identified regions-of-interest (false negatives) and thus increased the number of recordTarget, author, and informationRecipient errors.



	First Name	Last Name	SSN	Street Name	ZIP	Telephone Number	Organization
Gold Standard	593	630	114	276	304	184	108
Recognized Tokens	645	644	115	255	301	207	107
Correctly Recognized Tokens	502	545	109	245	289	143	107

Figure 4.16: Selected PHI elements (annotation)

The analysis of the annotation results provides a more detailed view where we evaluated the system's performance by calculating the precision, recall, and f-measures (cf. [20]) as follows: The precision metric is determined by x/X in the context of PHI class p where x is the number of correctly recognized class p tokens and X the total number of tokens recognized as p. Recall is defined as y/Y in the context of PHI class p where y is the number of correctly recognized class p tokens and Y the total number of class p tokens present in the document. To combine precision and recall, the f-measure is calculated as the harmonic mean as follows:

$$F = \frac{2 * precision * recall}{precision + recall} \tag{4.2}$$

As shown in Figure 4.16, PHI classes like the social security number, postcodes, and organization have reliably been identified due to their static structure (Austrian social security numbers combine a four-digit number with the birth date, while Austrian

postcodes are always comprised of four digits) and the availability of extensive lookup tables (organizations), leading to precision/recall values of 0.95/0.96, 0.96/0.95, and 1/0.99 respectively. Street names have also been reliably identified due to lookup tables, though some were misspelled or incomplete (missing 'Strasse', i.e. the German equivalent for 'Street' or 'Avenue'). Names and telephone numbers could reach only lower precision/recall values (0.78/0.85, 0.85/0.87, and 0.69/0.78): Apart from the issues described above, names were also miscategorized (first/last names were mixed up), whereas phone numbers suffered from character recognition errors, formatting issues (slashes, hyphens) and inconsistent domains with different digit counts (with/without area codes or call-through numbers).

As the transformation rules in the CDA body sections were largely comprised of content-based conditions, the remaining errors in the CDA sections (cf. figure 6) resulted from incomplete keyword sets (e.g., 'diagnosis', 'diagnostic results', etc.) or misspelled keywords due to OCR recognition errors. Other causes included keywords and indicators missing in the original document (e.g., 'to be sent to') or structural variations in the source documents. Finally, a large number of sections could not be successfully transformed because they simply did not exist in the original health document.

Given the fact that no special tweaking was done with the configuration and rules, the system produced very good results. Concerning the annotation phase, we achieved an overall f-measure of 91.5%, compared to 96.3% of in [123], 96.7% in [109], and 88.3% [54], but including OCR errors which are often neglected in other approaches where, for the sake of comparability, a standardized set of test documents is used (cf. i2b2 de-identification challenges). Still, the results also indicate that there is room for improvement, especially in the area of names, the primary identifiers of persons. The test run was conducted with a setting that was intended to keep a balance between precision and recall. A production system would lay the emphasis on recall for the de-identification purpose, as a false-positive is less critical than a false-negative. As each module relies on the quality of the outcome of the previous modules, emphasis has to be laid on the OCR and annotation phases. OCR quality is expected to increase with further document-type specific thresholds and configurations, i.e. optimizing the text box thresholds, using font-specific configurations, etc. To counter the first/last name mix up issue, the name categories could also be merged to a single PHI class, which would allow creating more general annotation conditions in the transformation rules. Further annotation improvement strategies include using more advanced rules (cascading compositions) and more greedy rules with explicit exceptions, as well as tweaking the confidence values. The integration of further external data sources can also improve the annotation quality by providing extended lookup tables and direct-match verification opportunities. With access to extended training data, the machine learning components should also prove useful.

Better OCR and annotation quality also positively affects the transformation, although the transformation rules are designed in such a way that they are able to compensate OCR and annotation shortcomings. The rules for the test run included simple rule compositions with few condition alternatives (structural and content-based). The rules

can be adapted to specifically address certain known OCR and annotation limitations, such as the first/last name issue or misspelled words by creating less-specific conditions (e.g., more general regex). The transformation module's CDA validator provides the necessary feedback to iteratively adapt the OCR/annotation/transformation configuration settings and rules to finally improve the overall result.

CHAPTER 5

Conclusion

5.1 Summary

Privacy is a commodity whose role has become more important than ever. Given the interconnectivity of computer systems and the large amounts of data produced every day, there is a need for the protection of data, especially sensitive personal data like health data. Since legal or organizational protection measures are often not sufficient, they need to be supplemented by technical solutions. These technical security measures should ensure that sensitive data is disclosed to authorized persons only. With regard to health data, authorized persons are in general patients and trusted health care providers. Obviously, health professionals should have access to their patients' data to provide the best possible health service, but patient data should not be disclosed to other interested parties to prevent adverse consequences such as discrimination against patients. However, personal health data is also valuable for research purposes to improve current practices in health care and also to investigate new research directions. Still, when making the data available to the research community, the patients' privacy has to be ensured. Therefore, a suitable security architecture for handling health data should be able to (i) grant access to authorized parties such as health care providers responsible for the treatment (which requires the patient's consent) and (ii) allow privacy-preserving secondary use for research.

In Chapter 2, this thesis first introduced information privacy and privacy-enhancing technologies and discussed their properties in seven distinctive dimensions which help to categorize different types of PETs. Then pseudonymization and its properties are investigated. While different pseudonymization approaches have been proposed in literature, they all focus on identity, provide unlinkability, and are reversible, which are useful properties for handling sensitive health data.

In Chapter 3, PERiMETER was introduced, a holistic security architecture based on pseudonymization which allows the concurrent use of health data for both direct care (primary use) as well as privacy-preserving research (secondary use). Its main

features are as follows: (i) The pseudonym-based access control relies on root and shared pseudonyms which represent access rights to health records. They hide which health records belong to a particular patient, unless the correct pseudonyms are known. (ii) These pseudonym links are protected by encryption with multiple keys that form a layer-based security architecture where the different layers are responsible for user authentication, access authorization, and data access to the health records. The security architecture also distinguishes between data owner, authorized users, and secondary users. Finally, (iii) PERiMETER was designed to be used with secure hardware to keep vital cryptographic keys secure during their use. Then, extensions to the basic pseudonymization method were described that deal with ensuring data integrity or allow for asynchronous operations. Furthermore, different application scenarios were described including the pseudonymization of plain documents or documents in the HL7 CDA format, as well as the integration of pseudonymization with electronic health records as exemplified with the Austrian Elektronische Gesundheitsakte.

In Chapter 4, pseudonymization was discussed as privacy-enhancing technology in a real life scenario involving paper-based archived health records that were to be made available for secondary use. This chapter in particular investigated the requirements of using the digitized paper-based records for privacy-preserving secondary use, including the proper de-identification of the records and their transformation into a standardized form. It then introduced MEDSEC, a methodology consisting of four phases: (i) The OCR phase converts the paper-based records into a machine-readable input string of characters and extends the strings with structural annotations. (ii) The annotation phase then extends the input text with annotations marking all identified elements of Protected Health Information. (iii) The transformation phase converts the unstructured input text into standardized CDA documents using the structural information and PHI annotations appended in the previous phases. And finally (iv) the pseudonymization phase ensures that the CDA documents are unlinkable so that the documents can be disclosed to the researchers.

5.2 Research questions revisited

In this section, the research questions stated in Section 1.3 are reviewed to investigate how they have been answered in this thesis.

Can pseudonymization be used as a holistic security architecture controlling authentication, authorization and data access to sensitive data?

PERiMETER is a pseudonym-based security methodology ensuring that health records are stored in pseudonymized form so that the data can be used for privacy-preserving secondary use, i.e. the records are unlinked from both the patient and other records. For primary use in direct health care, each patient acts as data owner and is able to create user- and record-specific access authorizations for trusted health care providers. These

access authorizations in the form of shared pseudonyms allow the authorized parties to reidentify the correct patient's records by having knowledge of the corresponding pseudonym links. With the correct pseudonyms, the correct patient's records can be retrieved. Each user is also assigned unique sets of cryptographic keys including the outer asymmetric keypair which is used to unambiguously identify and authenticate the user. In order to protect the outer private key, it is created and stored on a user-owned security token that acts as authentication device. The user-specific cryptographic keys also include the inner symmetric key which is used to protect the pseudonym links and ensures that only the key holder is able to decrypt the pseudonyms. As each user is provided with security tokens for authentication and general access to the pseudonymized data can be restricted to authenticated users only, PERiMETER represents a pseudonym-based holistic security architecture that handles authentication, authorization, and access control to sensitive data.

How does pseudonymization fare in a real-world scenario with unstructured source data? What are the preconditions for successful pseudonymization?

As demonstrated by MEDSEC, pseudonymization is suitable for privacy-preserving secondary use and, in contrast to anonymization, is reversible so that patients can also be reidentified. This is beneficial in cases where potential patients are selected for clinical trials or need to be contacted to provide further details about their medical histories. In order to handle unstructured source data, several requirements have to be met: At first, the source data has to be transformed into a machine-readable format since manual handling of the data is usually unfeasible due to the large amounts of records in a real-world scenario. While health record standardization has recently been promoted, the majority of records in health record storage archives, representing sizable data sources, is still paper-based even when digitized, i.e. stored as scanned images. Using techniques such as OCR, the text contents of the images can be read and converted into machine-readable text. The unstructured data then needs to be converted into a standardized document standard such as HL7 CDA to facilitate interoperability and processing. Since the CDA standard dictates the structure of the documents' contents, the relevant information has to be identified within the source data and extracted so that CDA-compliant documents can be created. Then the documents can be pseudonymized. An important precondition for successful pseudonymization is the diligent de-identification of the documents, including recognizing and removing all patient-identifying elements (PHI). The CDA document standard is particularly suitable for pseudonymization since it collects person-identifying information in the administrative header section while keeping the actual medical content in the body section. Thus, the documents can be separated into header and body sections to naturally separate identifying information from medical data. Still, the body section usually contains several PHI elements as well because the contents of the documents contain natural language texts, especially in clinical discharge letters. Therefore, the body sections also have to be scrubbed before they can be made

available for secondary use. The four phases of MEDSEC include all these steps and demonstrate how to convert and pseudonymize existing health data for privacy-preserving secondary use.

5.3 Benefits of the developed approach and comparison to related work

Compared to existing approaches, the pseudonymization framework presented in this thesis provides several benefits. Whereas pseudonymization has been used in approaches focusing on either primary or secondary use (cf. Section 2.4), PERiMETER's novelty lies in its design as a holistic security infrastructure combining authentication, authorization, and access control. This allows it to be used concurrently for both primary as well as secondary use. Furthermore, the pseudonym-based infrastructure also provides user-controlled security which does not depend on a dedicated trusted third party, unlike existing work (e.g., [85], [33], [60], [51], [43], [88], [1]). Since the patients' data is stored pseudonymized, the risk of data disclosure emanating from internal attackers can also be reduced because the correct secret cryptographic keys are required to revert pseudonymization. Therefore, any non-authorized party, including internal actors such as data administrators, does not have the ability to de-pseudonymize the data.

Similar to PERiMETER, the MEDSEC system represents a holistic approach for automatically converting and pseudonymizing medical records. Unlike existing systems which focus on individual aspects of the conversion process such as the de-identification of health records or the information extraction of relevant data (cf. Section 4.1), MEDSEC includes four distinctive phases to cover the individual requirements of converting paper-based archived health records into a standardized and pseudonymized form. Whereas other systems for de-identification (validated using test documents (cf. [119])) provide more accurate results (e.g., [123], [109]), the MEDSEC modules are designed to provide robust results in real-world scenarios. These modules working in conjunction help to mitigate known limitations of information identification and extraction such as incomplete source data, misinterpreted characters (OCR), or different data formats. Therefore, MEDSEC can be applied to a wider variety of health data compared to the more specialized existing systems presented in literature.

5.4 Limitations and future research directions

Whereas PERiMETER and MEDSEC demonstrate the practicability of pseudonymization when handling sensitive health data, there are still some limitations. In PERiMETER, pseudonymization is limited to health data-at-rest to hide the static links between patients and their health records, while the dynamic or temporal factor is neglected. That means that the links between health records can be reidentified when the order of retrieved record (fragments) and the time interval between the retrieval operations is tracked: Two records retrieved consecutively by a health care provider are likely to be related to the

same patient. Even when the connection between client and database server is properly protected by transport-layer end-to-end encryption (TLS) preventing eavesdropping, a malicious database administrator with access to the data retrieval logs is able to identify the access patterns and, therefore, to break the pseudonymization. We identify two research directions to prevent this problem: On the one hand, the returned result set can be modified so that the result must still be processed at the data querier's side before useful information can be extracted. Private information retrieval [31] hides the access pattern by either distributing sensitive data to multiple databases (multi-server PIR) or applying cryptographic operations (single-server PIR). However, since the result of a data query in this thesis is a complete health document, applying PIR-based solutions considerably hinders secondary use which is one of the requirements identified in this thesis. Future research in this direction might be able to answer the question if techniques such as PIR can be properly combined with pseudonymization. On the other hand, the access pattern can also be obscured by inserting fake data requests to hide the link between two legitimate data requests. In this case, the research question is how large the amount of fake requests has to be to sufficiently hide the access pattern. One privacy model that relies on the insertion of noise in data requests for privacy preservation is differential privacy [41]. The idea of differential privacy is to keep the changes in a query result due to a single change in the original data (inclusion of a particular data row representing the presence of a particular person in this data set) at a minimal manageable level. This can be ensured by adding sufficient random noise which is dependent on the maximum deviation of the query result set. For example, if the query is a simple count of the rows, the maximum deviation when adding the particular row is 1, which means that the required noise is also 1. Thereby, a differentially private query mechanism can hide the existence of a particular person in a data set, but in contrast to PERiMETER, the original database remains unchanged and the result sets are modified before they are sent to the data querier (or alternatively a subset of the original data or synthesized data set is created which produces similar results for any allowed query types). Differential privacy works with aggregation queries, but it is unclear if it can be combined with pseudonymization to handle exact match queries (or modify the queries to range queries) without further research. MEDSEC's PHI identification and pseudonymization mechanism is limited to the machine-readable text inputs produced in the OCR phase, while images like radiological files (DICOM), which are often part of health records, are neglected. Even when metadata fields are scrubbed in such a file, the image itself might contain PHI. Therefore, the automated PHI recognition process could be extended to include images using image recognition and processing approaches.

Bibliography

[1] H. Aamot, C. Kohl, D. Richter, and P. Knaup-Gregori. Pseudonymization of patient identifiers for translational research. *BMC Medical Informatics and Decision Making*, 13(75), 2013.

[2] R. Anderson, M. Bond, J. Clulow, and S. Skorobogatov. Cryptographic processors - a survey. Technical report, University of Cambridge, Computer Laboratory, 2005.

[3] A. Appari and M. E. Johnson. Information security and privacy in healthcare: Current state of research. *International Journal of Internet and Enterprise Management*, 6(4):279–314, 2010.

[4] D. E. Appelt and D. J. Israel. Introduction to information extraction technology. A Tutorial Prepared for IJCAI-99, August 1999.

[5] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30, 2006.

[6] J. Attridgde. An overview of hardware security modules. Technical report, SANS Institute, 2002.

[7] R. Au and P. Croll. Consumer-centric and privacy-preserving identity management for distributed e-health systems. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, pages 234–244, Washington, DC, USA, 2008. IEEE Computer Society.

[8] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, 2004.

[9] A. Beimel. Secret-sharing schemes: A survey. In *Proceedings of the Third International Conference on Coding and Cryptology*, IWCC'11, pages 11–46, Berlin, Heidelberg, 2011. Springer-Verlag.

[10] K. Benitez and B. Malin. Evaluating re-identification risks with respect to the HIPAA privacy rule. *Journal of the American Medical Informatics Association*, 17(2):169–177, 2010.

[11] J. J. Berman. Concept-match medical data scrubbing. How pathology text can be used in research. *Archives of Pathology and Laboratory Medicine*, 127(6):680–686, 2003.

[12] J. J. Berman. *Ruby Programming for Medicine and Biology*. Jones and Bartlett Publishers, 2008.

[13] M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In *Advances in Cryptology - Eurocrypt 98*, pages 127–144, 1998.

[14] M. K. Bond. *Understanding Security APIs*. PhD thesis, University of Cambridge, Computer Laboratory, Emmanuel College, 2004.

[15] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology - Crypto 2004*, pages 41–55, 2004.

[16] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *Advances in Cryptology - Eurocrypt 2004*, pages 506–522, 2004.

[17] H. Bouzelat, C. Quantin, and L. Dusserre. Extraction and anonymity protocol of medical file. In *Proceedings of the AMIA Annual Fall Symposium 1996*, pages 323–327, 1996.

[18] G. Brassard, C. Crepeau, and J.-M. Robert. All-or-nothing disclosure of secrets. In *Advances in Cryptology - Crypto 86*, pages 234–238, 1987.

[19] E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *Proceedings of the 11th ACM Converence on Computer and Communications Security*, pages 132–145, 2004.

[20] M. Buckland and F. Gey. The relationship between recall and precision. *Journal of the American Society for Information Science*, 45(1):12–19, 1994.

[21] J. Callas, I. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. OpenPGP Message Format. RFC4880, November 2007.

[22] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology - Eurocrypt 2001*, pages 93–118, 2001.

[23] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In *Advances in Cryptology - Crypto 97*, pages 410–424, 1997.

[24] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 21–30, New York, NY, USA, 2002. ACM.

[25] M. Campbell, C. Snowden, D. Francis, D. Elbourne, A. McDonald, R. Knight, V. Entwistle, J. Garcia, I. Roberts, and A. Grant. Recruitment to randomised trials: Strategies for trial enrollment and participation study: The STEPS study. *Health Technology Assessment*, 11(48):1–140, 2007.

[26] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky. Deniable encryption. In *Advances in Cryptology - Crypto 97*, pages 90–104, 1997.

[27] J. Caumanns. Der Patient bleibt Herr seiner Daten: Realisierung des eGK-Berechntigungskonzepts über ein ticketbasiertes virtuelles Dateisystem. *Informatik-Spektrum*, 29(5):323–331, 2006.

[28] D. Chaum. Untracable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[29] D. Chaum. Security with identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[30] D. Chaum. Group signatures. In *Advances in Cryptology - Eurocrypt 91*, pages 257–265, 1991.

[31] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science*, pages 41–51, 1995.

[32] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: context-driven automatic semantic annotation with C-PANKOW. In *Proceedings of the 14th International Conference on World Wide Web*, pages 332–341, 2005.

[33] B. Claerhout and G. DeMoor. Privacy protection for clinical and genomic data: The use of privacy-enhancing techniques in medicine. *International Journal of Medical Informatics*, 74:257–265, 2005.

[34] Council of Europe. *European Convention on Human Rights*. Martinus Nijhoff Publishers, 1987.

[35] G. D. Crescenzo, T. Malkin, and R. Ostrovsky. Single database private information retrieval implies oblivious transfer. In *Advances in Cryptology - Eurocrypt 2000*, pages 122–138, 2000.

[36] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters. *Developing Language Processing Components with GATE Version 6 (a User Guide)*. University of Sheffield, Department of Computer Science, April 2011.

[37] E. Damiani, S. D. C. di Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati. Balancing Confidentiality and Efficiency in Untrusted Relational DBMSs. In *ACM Conference on Computer and Communications Security*, pages 93–102. ACM, 2003.

[38] W. Diffie and M. Hellman. Privacy and authentication: An introduction to cryptography. *Proceedings of the IEEE*, 67(3):397–427, March 1979.

[39] R. Dingledine, M. J. Freedman, and D. Molnar. The Free Haven project: distributed anonymous storage service. In *International workshop on Designing privacy enhancing technologies*, pages 67–95, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[40] R. Dingledine, N. Mathewson, and P. Syverson. Tor: the second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium*, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[41] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer Berlin Heidelberg, 2006.

[42] ELGA GmbH. Die elektronische Gesundheitsakte. Online: http://www.elga.gv.at.

[43] B. Elger, J. Iavindrasana, L. Iacono, H. Mueller, N. Roduit, P. Summers, and J. Wright. Strategies for health data exchange for secondary, cross-institutional clinical research. *Computer Methods and Programs in Biomedicine*, 99(3):230–251, 2010.

[44] 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*, L 281:31–50, 1995.

[45] S. Even and O. G. G. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28:637–647, 1985.

[46] Federal information processing standards publication. Security requirements for cryptographic modules (FIPS PUB 140-2). Technical report, Institute of Standards and Technology (NIST), 2001.

[47] D. Ferrucci and A. Lally. Uima: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10:327–348, 2004.

[48] S. Fischer-Hübner. *IT-Security and Privacy: Design and Use of Privacy-Enhancing Security Mechanisms*. Springer, Berlin, 2001.

[49] Fraunhofer Institut. Spezifikation der Lösungsarchitektur zur Umsetzung der Anwendungen der elektronischen Gesundheitskarte, 2005.

[50] F. J. Friedlin and C. J. McDonald. A software tool for removing patient identifying information from clinical documents. *Journal of the American Medical Informatics Association*, 15:601–610, 2008.

[51] D. Galindo and E. R. Verheul. Microdata sharing via pseudonymization. Technical report, Joint UNECE/Eurostat work session on statistical data confidentiality, 2007.

[52] O. Goldreich. *Foundations of Cryptography.* Cambridge University Press, 2007.

[53] S. Grocer. Sony, Citi, Lockheed: Big Data Breaches in History. The Wall Street Journal, June 2011.

[54] C. Grouin, A. Rosier, O. Dameron, and P. Zweigenbaum. Testing tactics to localize de-identification. In *Medical Informatics in Europe conference (MIE'2009)*, pages 735–739, 2009.

[55] K. Grün, M. Karlinger, and M. Schrefl. Schema-aware labelling of XML documents for efficient query and update processing in SemCrypt. *International Journal of Computer Systems: Science & Engineering*, 21(1), 2006.

[56] K. Hara. Applying a SVM based chunker and a text classifier to the de-id challenge. In *i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data*, pages 1–13, 2006.

[57] Health Level 7 International. CDA Release 2. Retrieved from http://www.hl7.org, 2015.

[58] Health Level 7 International. Reference Information Model 2.41. Retrieved from http://www.hl7.org, 2015.

[59] B. Holcombe. *Government Smart Card Handbook.* U.S. General Services Administration (GSA), 2004.

[60] L. L. Iacono. Multi-centric universal pseudonymisation for secondary use of the EHR. *Studies in Health Technology and Informatics*, 126:239–247, 2007.

[61] Integrating the Healthcare Enterprise (IHE). IHE IT Infrastructure (ITI) Technical Framework 12.0. Retrieved from http://www.ihe.net, September 2015.

[62] Integrating the Healthcare Enterprise (IHE). IHE IT Infrastructure (ITI) Technical Framework Volume 1 (ITI TF-1) Integration Profiles, September 2015.

[63] T. Jurgensen and S. Guthery. *Smart Cards: The Developer's Toolkit.* Pearson Education, Inc. Prentice Hall PTR, Upper Saddle River, 2002.

[64] A. A. E. Kalam, Y. Deswarte, G. Trouessin, and E. Cordonnier. Smartcard-based anonymization. In *Proceedings of the Sixth International Conference on Smart Card Research and Advanced Applications (CARDIS)*, pages 49–66, 2004.

[65] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 364–373, 1997.

[66] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *IEEE 23rd International Conference on Data Engineering (ICDE2007)*, pages 106–115, 2007.

[67] S. Liu and R. Kuhn. Data loss prevention. *IT Professional*, 12(2):10–13, 2010.

[68] G. Lowe. An attack on the Needham-Schroeder public-key authentication protocol. *Inf. Process. Lett.*, 56(3):131–133, 1995.

[69] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1):3, 2007.

[70] S. Marsh and M. R. Dibben. Trust, untrust, distrust and mistrust - an exploration of the dark(er) side. In *Proceedings of the Third International Conference on Trust Management*, iTrust'05, pages 17–33, Berlin, Heidelberg, 2005. Springer-Verlag.

[71] U. Maurer. Information-theoretic cryptography. In *Advances in Cryptology - Crypto 99*, pages 47–65, 1999.

[72] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Crypography*. CRC Press, 2001.

[73] M. Murugesan and C. Clifton. Providing privacy through plausibly deniable search. In *Prooceedings of the 2009 SIAM International Conference on Data Mining*, pages 768–779, 2009.

[74] S. M. Mystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle. Extracting information from textual documents in the electronic health record: A review of recent research. *IMIA Yearbook of Medical Informatics*, 1:128–144, 2008.

[75] National Partnership for Women & Families. Faces of genetic discrimination, July 2004.

[76] I. Neamatullah, M. Douglass, L. Lehman, A. Reisner, M. Villarroel, W. Long, P. Szolovits, G. Moody, R. Mark, and G. Clifford. Automated de-identification of free-text medical records. *BMC Medical Informatics and Decision Making*, 8:1–17, 2008.

[77] NEMA. Digital imaging and communications in medicine. Open Standard, 2015.

[78] R. B. Ness. Influence of the HIPAA privacy rule on health research. *Journal of the American Medical Association*, 298(18):2164–2170, 2007.

[79] R. Noumeir, A. Lemay, and J.-M. Lina. Pseudonymization of radiology data for research purposes. *Journal of Digital Imaging*, 20:284–295, 2007.

[80] Office for Civil Rights. Guidance regarding methods for de-identification of protected health information in accordance with the health insurance portability and accountability act (HIPAA) privacy rule, 2012.

[81] A. O'Neill, C. Peikert, and B. Waters. Bi-deniable public-key encryption. In *Advances in Cryptology - Crypto 2011*, pages 525–542, 2011.

[82] R. L. Peterson. Encryption system for allowing immediate universal access to medical records while maintaining complete patient control over privacy. *US Patent US 2003/0074564 A1*, 2003.

[83] A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology (v0.34). Online: http://dud.inf.tu-dresden.de/Anon_Terminology.shtml, 2010.

[84] C. P. Pfleeger and S. L. Pfleeger. *Security in Computing*. Prentice Hall, fourth edition, 2006.

[85] K. Pommerening. Medical Requirements for Data Protection. In *Proceedings of IFIP Congress, Vol. 2*, pages 533–540, 1994.

[86] K. Pommerening and M. Reng. Secondary Use of the EHR via Pseudonymisation. *Medical Care Compunetics 1, IOS Press*, pages 441–446, 2004.

[87] M. O. Rabin. How to exchange secrets with oblivious transfer. Technical Report TR-81, Aiken Computation Lab, Harvard University, 1981.

[88] Y. Rahim, S. Sahib, and M. Ghani. Pseudonymization techniques for privacy study with clinical data. *International Journal of Transformation Science*, 2(6):75–78, 2012.

[89] Regenstrief Institute. Logical observation identifiers names and codes (LOINC), 2008.

[90] Republic of Austria. Datenschutzgesetz 2000 (DSG 2000), BGBl. I Nr. 165/1999, 1999.

[91] A. Rezgui, A. Bouguettaya, and M. Y. Eltoweissy. Privacy on the web: Facts, challenges, and solutions. *IEEE Security & Privacy*, 1(6):40–49, 2003.

[92] B. Riedl, V. Grascher, S. Fenz, and T. Neubauer. Pseudonymization for improving the privacy in e-health applications. In *Proceedings of the 41st Hawai'i International Conference on System Sciences*, 2008.

[93] B. Riedl, V. Grascher, and T. Neubauer. Applying a threshold scheme to the pseudonymization of health data. In *Proceedings of the 13th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'07)*, 2007.

[94] B. Riedl, V. Grascher, and T. Neubauer. A secure e-health architecture based on the appliance of pseudonymization. *Journal of Software*, 2008.

[95] B. Riedl, T. Neubauer, G. Goluch, O. Boehm, G. Reinauer, and A. Krumboeck. A secure architecture for the pseudonymization of medical data. In *Proceedings of the Second International Conference on Availability, Reliability and Security*, pages 318–324, 2007.

[96] S. Rosenblatt. 13 Revelations From The Sony Hack. CNET, December, 13 2014.

[97] T. Rössler, H. Leithold, and R. Posch. E-voting: A scalable approach using XML and hardware security modules. In *Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EFF'05)*, 2005.

[98] RSA Laboratories. PKCS#11 v2.20: Cryptographic token interface standard, 2004.

[99] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. on Knowl. and Data Eng.*, 13(6):1010–1027, Nov. 2001.

[100] S. Sarawagi. Information extraction. *Found. Trends Databases*, 1:261–377, March 2008.

[101] M. Schrefl, J. Dorn, and K. Grün. SemCrypt - Ensuring Privacy of Electronic Documents through Semantic-based Encrypted Query Processing. In *Proc. Int. Workshop on Privacy Data Management (PDM 2005)*. IEEE Computer Society Press, 2005.

[102] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[103] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 27:379–423 and 623–656, 1948.

[104] J. Silver-Greenberg, M. Goldstein, and N. Perlroth. JPMorgan Chase Hacking Affects 76 Million Households. The New York Times, October 2, 2014.

[105] S. R. Simon, J. S. Evans, A. Benjamin, D. Delano, and D. W. Bates. Patients' attitudes toward electronic health information exchange: Qualitative study. *Journal of Medical Internet Research*, 11(3), 2009.

[106] H. J. Smith. Privacy policies and practices: Inside the organizational maze. *Commun. ACM*, 36(12):104–122, Dec. 1993.

[107] D. Solove. A taxonomy of privacy. *University of Pennsylvania Law Review*, 154(3):477–564, 2006.

[108] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[109] G. Szarvas, R. Farkas, and R. Busa-Fekete. State-of-the-art anonymization of medical records using an iterative machine learning framework. *Journal of the American Medical Informatics Association*, 14:574–580, 2007.

[110] R. Taira, A. Bui, and H. Kangarloo. Identification of patient name references within medical documents using semantic selectional restrictions. In *Proceedings of AMIA Symposium 2002*, pages 757–761, 2002.

[111] C. Thielscher, M. Gottfried, S. Umbreit, F. Boegner, J. Haack, and N. Schroeders. Data processing system for patient data. *Int. Patent, WO 03/034294 A2*, 2005.

[112] United Nations General Assembly. Universal declaration of human rights. Retrieved from http://www.un.org/en/documents/udhr/, 1948.

[113] US Congress. Privacy act of 1974, pub.l. 93-579. 88 stat. 1896. Retrieved from http://www.gpo.gov/fdsys/pkg/USCODE-2012-title5/pdf/USCODE-2012-title5-partI-chap5-subchapII-sec552a.pdf, 1974.

[114] US Congress. Uniting and Strenghtening America by Providing Appropriate tools Required to Intercept and Obstruct Terrorism (USA PATRIOT Act) Act of 2001, Pub. L. No. 107 - 56. 115 Stat. 272. Retrieved from http://www.gpo.gov/fdsys/pkg/PLAW-107publ56/pdf/PLAW-107publ56.pdf, 2001.

[115] US Congress. Genetic information nondiscrimination act, pub.l. 110-233, 122 stat. 881. Retrieved from https://www.govtrack.us/congress/bills/110/hr493/text, 2008.

[116] US Department of Health & Human Services, Office for Civil Rights. Summary of the HIPAA Privacy Rule. Online, 2003.

[117] US Supreme Court. Wheaton v. Peters, 33 U.S. 8 Pet. 591 591. Retrieved from https://supreme.justia.com/cases/federal/us/33/591/case.html, 1834.

[118] US Supreme Court. Olmstead v. United States, 277 U.S. 438. Retrieved from https://supreme.justia.com/cases/federal/us/277/438/case.html, 1928.

[119] O. Uzuner, Y. Luo, and P. Szolovits. Evaluating the state-of-the-art in automatic de-identification. *Journal of the American Medical Informatics Association*, 14(5):550–563, 2007.

[120] H. C. Van Tilborg and S. Jajodia. *Encyclopedia of cryptography and security*. Springer Science & Business Media, 2011.

[121] S. Velupillai, H. Dalianisa, M. Hassela, and G. H. Nilsson. Developing a standard for de-identifying electronic patient records written in Swedish: Precision, recall and f-measure in a manual and computerized annotation trial. *International Journal Of Medical Informatics*, 78:19–26, 2009.

[122] D. Weerasinghe, M. Rajarajan, K. Elmufti, and V. Rakocevic. Patient privacy protection using anonymous access control techniques. *Methods of Information in Medicine*, 47:235–240, 2008.

[123] B. Wellner, M. Huyck, S. Mardis, J. Aberdeen, A. Morgan, L. Peshkin, A. Yeh, J. Hitzeman, and L. Hirschman. Rapidly retargetable approaches to de-identification in medical records. *Journal of the American Medical Informatics Association*, 14:564–573, 2007.

[124] A. F. Westin. *Privacy and Freedom*. Atheneum, 1967.

[125] D. C. Wherry. Secure your public key infrastructure with hardware security modules. Technical report, SANS Institute, 2003.

[126] D. J. Willison, K. Keshavjee, K. Nair, C. Goldsmith, and A. M. Holbrook. Patients' consent preferences for research uses of information in electronic medical records: Interview and survey data. *British Medical Journal*, 326:373, 2003.

[127] World Health Organisation. International statistical classification of diseases and related health problems (ICD). Standard, 2015.

[128] K. Zetter. Diginotar files for bankruptcy in wake of devastating hack. Wired, September 2011.

[129] N. Zhang, A. Rector, I. Buchan, Q. Shi, D. Kalra, J. Rogers, C. Goble, S. Walker, D. Ingram, and P. Singleton. A linkable identity privacy algorithm for HealthGrid. In *Proceedings of Healthgrid 2005*, pages 234–245, 2005.