

Privacy Preserving Data Collection

Swiss Federal Section For Cryptography

Rainer Baumann, Marcel Rieser, Reto Strobl, Francois Weissbaum, Andrea Weisskopf

rainer.baumann@ethz.ch, francois.weissbaum@gst.admin.ch

September 11, 2005

Abstract

Detailed data collection of individuals is a fundamental requirement for profound statistical analysis of a complex system as the health care system of Switzerland. The aim of this report is to model a privacy-preserving data collection system, to analyze an already proposed implementation and to develop a new proposal giving highest security.

Contents

1	Introduction	3
2	Problem Statement and Background	3
3	Privacy Preserving Data Collection	4
3.1	System Model	4
3.2	Privacy Preserving Data Collection	6
3.3	An out-of-the-model Attack	8
4	Patient Identifier	8
5	A Scheme Providing Weak Anonymity	9
5.1	An Informal Overview	9
5.2	Verification of Requirements	11
6	A Scheme Providing Strong Anonymity	12
6.1	An Informal Overview	12
6.2	Technical Details	13
6.3	Verification of Requirements	15
6.4	Offline Recovery Authority	17
7	The Strong Scheme in Practice	17
7.1	System Architecture	17
7.2	Performance Estimations	18
8	Further Applications and Extensions	21
8.1	Medical Insurance Card	21
8.2	Payment and Controlling	21
8.3	Balance of Risk	21
9	Discussion and Conclusion	21

1 Introduction

For understanding the uncontrolled growth of expenses of the Swiss health care system, a detailed privacy-preserving data collection is required. For this, we first present a system model and its requirements followed by two schemes implementing different levels of anonymity. Then we show a possible realizations of a scheme in practice.

In our system model of a privacy-preserving data collection we distinguish the following entities. *Sources* send their data to a *data collector* which stores them in a database. These data can be exported for statistical research by *data analysts*. In case of emergency a *recovery authority* is able to reveal an identity. For our system model we propose the following set of requirements: linkability, anonymity, recoverability, integrity, authenticity, and completeness.

A scheme published in 1997 by the Swiss Federal Statistics Office has several flaws. It gives unlimited potency to the data collector because he has control over the encryption process as well as over the database. The scheme also lacks any authentication and exported data was not explicitly protected.

In this report, we propose a sophisticated scheme and a possible system architecture. Our scheme solves the mentioned flaws: guaranteeing anonymity by separating the recovery agency from the data collector, authenticating data over all stages, and implementing explicit data exporting employing privacy-preserving algorithms.

The rest of this report is structured as follows. In section 2 the problem statement and some background information is given. In section 3 we introduce our general model and define some notions. Following, in section 5 and 6 we present two possible schemes, one implementing weak and one strong anonymity. Then, in section 7, we give a possible realization of the scheme guaranteeing strong anonymity. In section 8 we present further applications and extensions. Finally the discussion and conclusion can be found in section 9.

2 Problem Statement and Background

The Swiss people is proud of its excellent health care system. Unfortunately its cost runs out of control. In 1960 the costs added up to 5% of the swiss gross domestic product (GDP). A tremendous growth raised them up to 11,5% of the GDP or SFr. 50 billiards in 2003. In the last decade this growth was even accelerated. Figure 1 shows the disparity of growth between the GDP and the expense for health care between 1995 and 2003.

To figure out the reasons of this unintentional growth, the Swiss Federal Statistics Office (SFS) collected statistical data in several areas of the health care system, mainly general profiles. These data allowed for creating a good overall view and in addition provided several detailed perceptions for several areas. But they do not allow to trace in detail treatments, their cost and their success for a certain disease. Based on the findings from the statistics, various different counter measures were proposed and realized. Unfortunately they did not make it to confine the continuous growth. According to an OECD analysis there are two major factors for the continuous growth: the large offer and the distortion of the price formation. For studying these effects and the cost efficiency of a treatment, it lacks a profound data collection with detailed traces about individuals. But collecting detailed data about individuals is very critical due to privacy concerns.

The aim of this report is to model and develop a privacy-preserving data collection system covering the whole health care system of Switzerland.

An additional, very critical requirement specified by the SFS for such a system is, that it must be possible to resolve the identity of a specific person if some findings from the data collection are very essential for this person. By definition, this requirement introduces some weakness into every possible model since it requires an instances which can break the whole system (for more details see section 3.3).

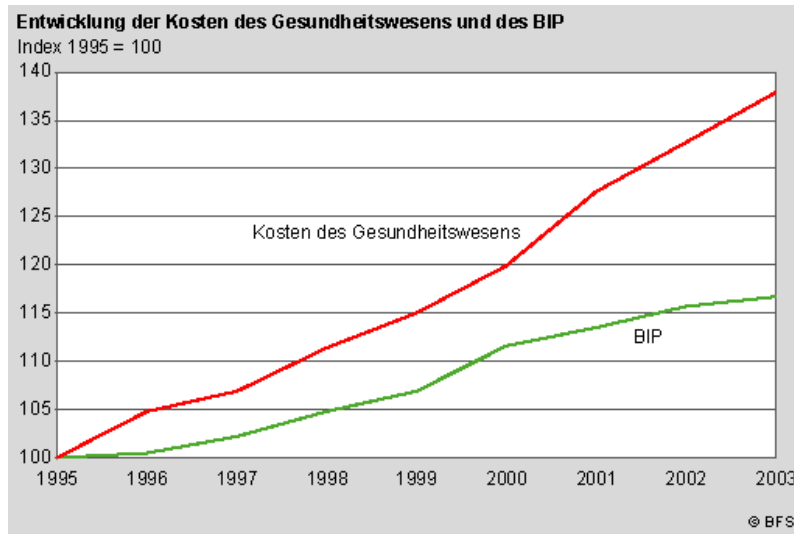


Figure 1: Development of cost of the Swiss health care system in percent compared to the GDP [2]

3 Privacy Preserving Data Collection

In this section, we explain what we mean by privacy-preserving data collection. In particular, we define the involved entities, the data-flow among these entities, and the properties that need to be guaranteed, as for example anonymity. Yet, we do not describe a specific implementation of such a system. This is the subject of following sections.

3.1 System Model

Our system model consists of a set of *sources*, a central *data collector*, several *data analysts*, and of a central *recovery authority*. On a high-level, the relations among these entities can be summarized as follows. The sources are applications that provide the medical data to be collected. The data collector gathers this data from the sources, and makes it available to data analysts which perform the desired statistical data analysis. The system ensures that personal identifying data is replaced with pseudonyms once it leaves the sources, and that only the recovery authority is capable of linking a pseudonym to its respective identity.

Figure 2 illustrates the high-level overview of the various entities and their relation in our system model. It also labels the protocols that determine the data flow among the entities. In the following, we describe the components and protocols in more detail.

Sources. The sources model client-applications that run at the institutes which generate the data to be collected. Such institutes could be for example hospitals, or insurance companies. We assume that every institute provides its data to its corresponding source application as a set of (non-anonymized) data records, each one representing a billable medical unit.

We assume that every data record is a tuple $(ID, data)$ consisting of a privacy-critical part ID describing the personal identification information of the patient, and of actual payload $data$ describing the medical unit (description, tax code, price, etc.). Deriving a unique identifier ID for a patient from personal identifying information is a hard task in itself. Section 4 proposes an algorithmic solution that should work well in practice.

We remark that the payload may include personal identification information of the treating doctor, or the treating nurses, and so on. In practice, it may be desirable to protect their privacy as

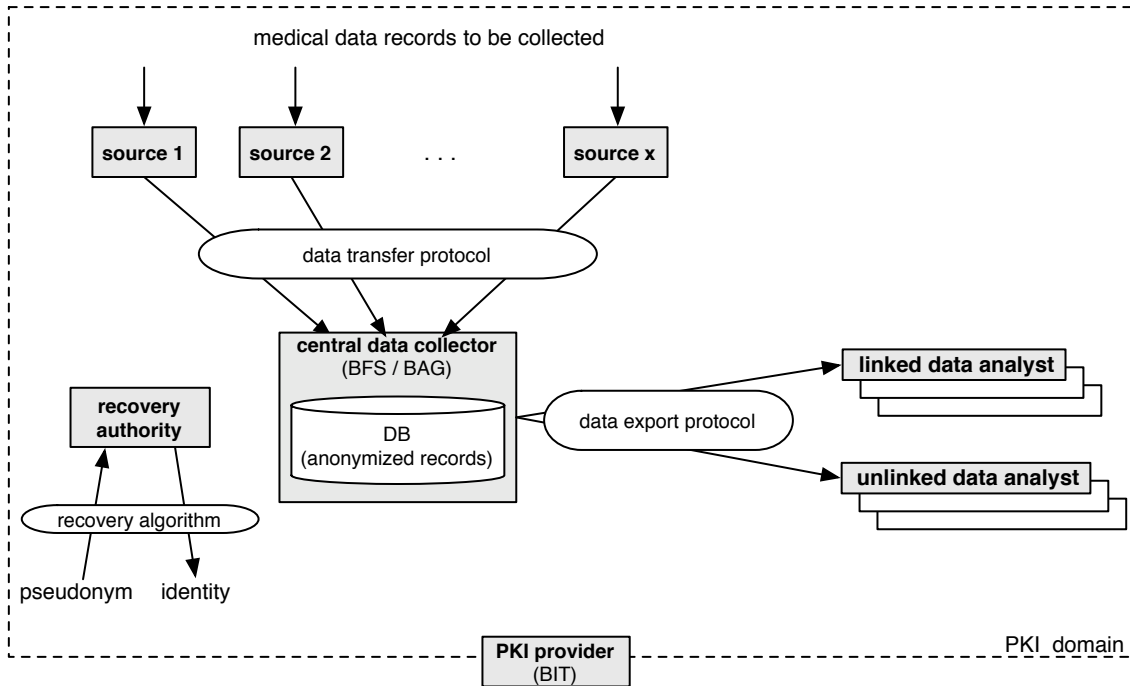


Figure 2: Overview of the System Model. Grey boxes represent entities, ovals represent the protocols involved, and arrows represent the data flow.

well. For simplicity, we do not address this explicitly here. Yet, we note that one can use the exact same techniques to anonymize these identities as we use to anonymize the identity of the patient. Notice also that in practice, it will be a tedious task to make all institutes provide their data in a common format, as it requires individual adjustments for every institute.

Central Data Collector. The central data collector is the entity that gathers an anonymized version of the data records that the sources collect. In the anonymized records, the identifier ID is replaced by a random string P , which we also call a *pseudonym*. No entity but the recovery manager must be able to link a pseudonym P to its corresponding identity ID .

This transfer as well as the anonymization is achieved through a so-called *data transfer protocol* that is run between every source and the data collector. To ensure proper anonymization, the protocol may also involve the recovery authority.

In practice, the central data collector could be for example the Swiss Federal Statistics Office, or the Swiss Federal Health Office.

Data Analysts. Data analysts are third-parties such as universities, state departments, insurance companies, etc., which analyze the collected information. We assume that there is a well-defined process on how data analysts can request a set of records from the data collector, and a law defining when such requests are valid and may be served. The data collector is responsible to ensure the enforcement of these laws. Once it has decided to serve a request, it transfers the requested data records to the data analyst through a so-called *data export protocol*. During this transfer, a pseudonym P may either stay the same, or be replaced with a 'fresh' pseudonym P' , or may be omitted completely.

We call a data analyst that receives records *with* pseudonyms a *linked-data analyst*, as he can reconstruct the medical history of a particular patient. On the other hand, we call a data analyst that

receives records without pseudonyms an *unlinked-data analyst*.

Recovery Authority. There may be situations, where the identity of a certain patient (or doctor, etc.) has to be recovered from a pseudonym. In our model, this can be done by a so-called *recovery algorithm*, which is run by the recovery authority. On input of a pseudonym P , the algorithm outputs the respective identity ID . We assume that in practice, there is law defining the circumstances under which such a recovery is legitimate. One has to trust the recovery authority that it only conducts legitimate recoveries.

It is crucial for the security of the system that one builds the recovery authority in a high-security environment, with limited access by people and programs.

In practice, it may also be desirable to have an independent entity that is responsible for the invocation of the recovery algorithm, and which keeps track of whom recovered whom on a public list. This is to make the anonymity as transparent as possible to the affected individuals.

Public Key Infrastructure. We assume that there is a public key infrastructure (PKI) that allows to authenticate sources, the data collector, the recovery authority, and the data analysts.

In practice, such a PKI could be managed for example by the Swiss Federal Office of Information Technology (BIT). The involved keys could be distributed by smart-cards, or secure USB sticks.

The Network. We assume that all entities can communicate with the central data collector through authentic reliable links. Specifically, if some entity sends a message m to the data collector, then the network guarantees that the data collector eventually receives m .

In practice, one can establish such a network by communicating over SSL/TLS connections with client and server authentication. The necessary certificates could be stored for example on the smart-cards of the PKI that we assume.

3.2 Privacy Preserving Data Collection

The system model so far describes the general working of a system for privacy-preserving data collection. Yet, it does not specify the concrete implementation of the basic components, which are the *data transfer protocol*, the *data export protocol*, and the *recovery algorithm*.

We call a concrete implementation of these components a privacy-preserving data collection *scheme*. Having this abstract notion of a scheme allows us to describe the desired properties of such a scheme in an implementation-independent way.

In the following, we discuss informally these properties of a privacy-preserving data collection scheme. This will serve as the basis for the analysis of the proposed solutions.

Linkability. For statistical purposes, it is desired that the data collector can link pseudonyms that originated from the same patient to each other (without knowing the actual identity of the patient). We can express this requirement in terms of the following relation between the records that the sources send and the records that the data collector stores in its DB:

Consider two sources, each one transferring a record $(ID, data)$ and $(ID', data')$, respectively, to the data collector. If ID and ID' are the same, then the data collector will store both records under the same pseudonym.

We should keep in mind that linkability induces an anonymity problem: If the link between a pseudonym and an identity is discovered once, then the complete history of this patient is revealed.

One particular example where this introduces bad side-effects is a source and a data analyst that pool their information. By matching records of the analyst with records of the source using the message payload, one can derive pseudonym to identity relations, and thereby, recover the personal medical history of any patient treated by this source to the horizon of the data hold by the analyst.

This is bad because a source should only be responsible for the anonymity of the records that it collects itself, and not about records collected by other sources. Linkability breaks with this natural requirement: If a source fails to keep the data of one of its patient secret, then not only this sources data about this patient is de-anonymized, but also the data collected by other sources on this patient.

Anonymity. There are two natural levels of anonymity a patient would expect: On the first level, he wants to remain anonymous with respect to data analysts, i.e., a data analyst must not be able to deduce a patients identity given only the patients anonymized records. On the second level, the patient wants to remain anonymous with respect to the data collectors. This is to avoid a big-brother scenario, where the central data collector (i.e., the state) keeps files for individuals. We summarize these properties below as weak and strong anonymity, respectively:

Consider two patients ID and ID' both treated at sources $\mathcal{S} = \{S_i, \dots, S_j\}$, and let P and P' denote their respective pseudonyms.

weak anonymity: If all sources in \mathcal{S} and the data collector follow the protocol, then no coalition of sources not in \mathcal{S} and data analysts can distinguish $(ID, P), (ID', P')$ from $(ID, P'), (ID, P)$ with a probability greater than $1/2$.

strong anonymity: If all sources in \mathcal{S} follow the protocol, then no coalition of the data collector, sources not in \mathcal{S} , and the data analysts can distinguish $(ID, P), (ID', P')$ from $(ID, P'), (ID, P)$ with a probability greater than $1/2$.

Notice that strong anonymity implies weak anonymity, as the data analysts get their data from the data collector. Notice also that the above definitions already take linkability into account. Specifically, they only require anonymity of a patient if all sources that treated this patient keep their data secret. If linkability was not a requirement, we could strengthen the above requirements in the sense that the anonymity of the data collected by a certain source is provided as long as the collecting source keeps the data secret.

Recoverability. In certain cases, it may be desirable to recover the identity of a patient from its pseudonym. This could be the case, for example, if a data analyst discovers a pattern indicating a disease that requires immediate treatment, it would be in the interest of the patient to trade his anonymity for health. In other words:

Let $(P, data)$ denote a record held by either the data collector or a data analyst, and suppose the pseudonym P originated from a patient ID . If the recovery authority invokes the recovery algorithm with input P , then it outputs ID .

Integrity. A fundamental requirement of a data collection scheme is that the data collector as well as the data analysts receive authentic records, i.e., records that have been inserted by a source.

We formulate this in terms of the following relation between records that sources send and the records that the data collector (or data analysts) receive.

If the data collector (or a data analyst) receives a record $(P, data)$ from the data transfer protocol (or the data export protocol, respectively), then the following holds:

1. the recovery authority outputs some identity ID on input P ,
2. there exists a source S that used the record $(ID, data)$ as input to the data transfer protocol.

Notice that integrity only guarantees that every record in the database has been provided by some source. Yet, it does not guarantee that every record represents a medical treatment unit that actually has occurred. In other words, we cannot

3.3 An out-of-the-model Attack

The requirements given in the previous section seem to provide a comprehensive protection of the patient's anonymity. Yet, even if a scheme satisfies all these requirements, there are still an attack that is not captured by these security properties.

The problem is that one cannot verify if the records provided by a source represent medical treatment units that actually have occurred. In other words, a source could invent (or could be fed with invented) records, which would then end up in the database just like the genuine records.

To see how this leads to a problem, suppose a source S invents a record $(ID, data)$ for a patient ID that it does not treat. According to our anonymity requirement, this source should not be able to learn the pseudonym(s) of this patient, even if it colludes with the data analysts or the data collector. However, since the invented record will end up in the database just like any other genuine record, S can compare the payload $data$ of the invented record with the payload of all records in the database. If it chose the payload uniquely enough, it will match the payload of only a small set of records from the data collector's database. This gives only a small set of possible pseudonyms for the identity ID . Repeating this step with another record will reduce the number of candidate pseudonyms gradually to one.

(Notice that the same attack can be done by a source that actually treats the patient ID . Yet in this case, we explicitly stated that the source must be trusted in order for anonymity to be guaranteed.)

In summary, as long as we do not have a notion of *genuine* data records, and a way to verify this property, no data collection scheme can ensure either weak or strong anonymity.

A natural definition of a *genuine record* could require that the record must be signed by the patient, i.e., by the individual whose anonymity could be compromised. This is only feasible if every patient can be authenticated by our PKI. In practice, this mean that every patient needs a smart-card (or a similar hardware token) issued by the PKI provider that is capable of generating a digital signature. Currently, this is clearly not the case in practice. However, this will change with the introduction of the Swiss medical card (see Section 8). We suggest to reconsider this approach when this card is introduced.

For the purpose of this work, we ignore this problem and assume that all records are genuine. In practice, one could somewhat raise the bar for inserting bad records by not only requiring not only the signature of a source to prove authenticity of a data record, but to associate with every source one or more individuals whose signature is also required; such individuals can be seen as 'data security officers' of the source institutions, and could be held responsible for forgeries.

4 Patient Identifier

In this section we discuss the necessary properties of a unique identifier and several possible implementations. They show that a medical card number would be the best choice but that a combination of common personal attributes is also a reasonable possibility.

As introduced in section 3.1, the privacy-preserving data collection system has to deal with many different data sources. For combining them to a consistent overall picture, it is necessary that all data sets of a person include the same unique identifier, a ID . There are several possibilities how to create such a ID as for example:

1. Postal address
2. Social security number

3. Combination of common attributes as first name, family name, date of birth, gender
4. Medical card number

Before discussing the listed possibilities, let us quickly define several requirements for the identifier. It should be

1. unique.
2. available for everybody (neonate, foreigner with permanent residence, ...).
3. not change over the whole life of an individual.
4. not obviously contain personal data.
5. known by every data source.

The postal address is not suitable since it contravenes at minimum with 2 to 5. At a glance, the social security number seems to be a quiet good choice. But when we check it with our requirements, it infringes upon number 2, 4 and 5. A combination of common attributes has also some problems. It violates the requirements 1, 3 and 4. But the violation of 4 can be solved by applying a cryptographic hash function as described in [5] and the probability for a violation of 1 can be kept quiet low [1]. If well designed, a medical card number is the perfect solution since it fulfills all requirements but it is not yet available.

Recapitulating, a medical card number would be the best solution but it is not yet available. So we propose for the time being to use a combination of common attributes as described in [5] but to switch as soon as possible to a medical card number.

To get an idea on how unique *IDs* according to [5] will be, we briefly summarize an experimental study [1]. When calculating *ID* for over 220 000 individuals, 0,3% false-positives were found. This means that 0,3% of the data would not reflect the medial history of real individuals, but the combined history of at least 0,6% of all individuals (in the case that each false-positive record consists of the data of two persons. If sometimes three or more individuals' data are merged because of the same *ID*, the percentage will be even higher). With a population of roughly 7 million individuals, the data of more than 42 000 individuals would be combined into 21 000 records. This fact should always be kept in mind when exceptional results are found in some statistics.

5 A Scheme Providing Weak Anonymity

In 1997, the federal office for statistics proposed a privacy-preserving data collection scheme [1]. It was intended to be used to collect statistical data of in-patient treatments. The scheme is similar to solutions used in France and other European countries. In the following, we summarize the proposed implementation and analyze it with respect to the aforementioned requirements.

5.1 An Informal Overview

In this section, we give an informal overview of the proposed scheme. For technical details, we refer to [1].

On a conceptual level, the scheme assumes a simpler system model than the one we presented. In particular, it only considers two types of entities, the sources and the central data collector. The recovery authority as well as data analysts are not considered as explicit entities of the model. Rather, these entities are an implicit part of the data collector.

The Data Transfer Protocol. Let $h(\cdot)$ denote a deterministic one-way function, that maps identities ID on (quasi)random bit-strings \tilde{P} (one-way means that it is impossible to find for a given \tilde{P} the corresponding ID). Consider a source S holding a medical record $(ID, data)$. The basic idea of the data transfer protocol is as follows (see Figure 3):

1. The source computes a temporary pseudonym \tilde{P} as $h(ID)$, and sends $(\tilde{P}, data)$ encrypted to the data collector.
2. Upon receiving this message, the data collector decrypts it and checks if it has already given out a pseudonym P for \tilde{P} . If not, it chooses the pseudonym P at random, and remembers the matching (\tilde{P}, P) .
3. The data collector stores the tuple $(P, data)$ in its database.

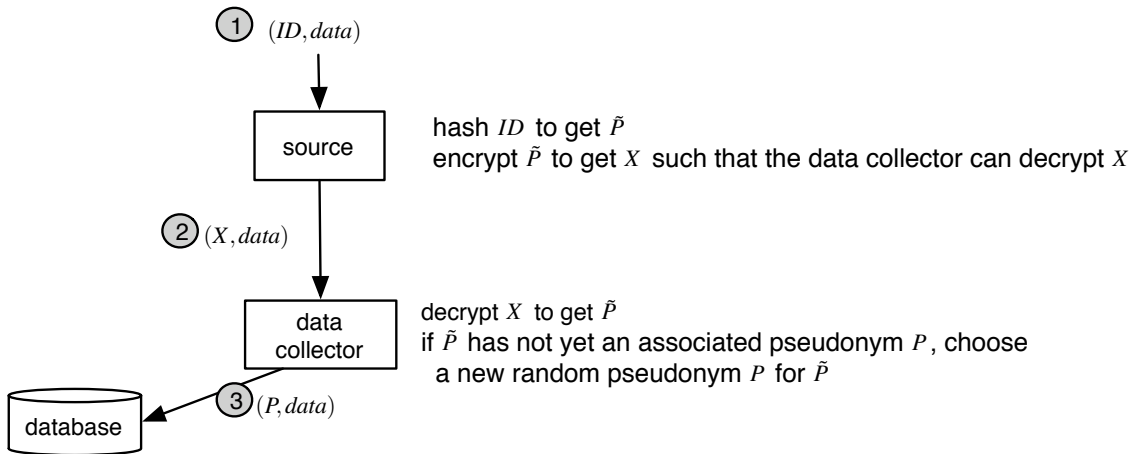


Figure 3: Data Flow in the Data Transfer Protocol

Technically, choosing a random pseudonym P and remembering the matching (P, \tilde{P}) is done by choosing a secret key k at system initialization, and then computing the pseudonym P as the symmetric encryption $ENC_k(ID)$ of ID , using for example the IDEA [9] or the AES encryption scheme.

Using this technique, the secret key k becomes the key to recover identities of a pseudonym, and thus, deserves special protection. To address this issue, the authors suggest to share the key k among say three individuals, who would have to pool their key-shares whenever the key is needed.

The Data Export Protocol. As mentioned above, this scheme does not have the concept of data analysts, but assumes that all analysis is done directly on the database of the data collector. This introduces some security issues, as the analysts have access to *all* collected records, and not only on a subset of the records.

The scheme mentions that it would be possible to give data to external analysts, such as universities, or health care providers. These external analysts are a similar concept as the data analysts in our system model. Yet, to export data, the scheme does not foresee a special protocol, but simply dumps the data directly from the data collector's database.

As a consequence, external analysts that perform regular analyses (say once every year), will gradually get a bigger picture on the individual medical histories. To address this issue, the scheme proposes to refresh the pseudonyms in the database periodically (for example, once a year), so that linkage of records cannot be done across multiple years.

To do this refresh, the data collector computes for every pseudonym P in the database the corresponding \tilde{P} , chooses a fresh pseudonym P' , and exchanges in the database all pseudonyms P with P' . Furthermore, it remembers the new matching (P', \tilde{P}) for future updates.

Technically, the refresh is done by first decrypting all pseudonyms P with key k . This gives all temporary pseudonyms \tilde{P} . Then, the collector chooses a fresh key k' , and computes the fresh pseudonyms P' as the symmetric encryption of the \tilde{P} 's under the fresh key k' .

The Recovery Algorithm. The recovery algorithm in this scheme is conducted by the data collector and the source together. To resolve the identity behind a pseudonym P , the data collector first looks up the corresponding temporary pseudonym \tilde{P} . It then asks a source for a list of all its patients, and checks for every ID' if $\tilde{P} = h(ID')$. If this holds for some ID , then this is the sought identity.

5.2 Verification of Requirements

It is easy to see that the scheme satisfies linkability, recoverability, and weak anonymity. While this provides already some protection of the privacy of the involved patients, there are still some issues that remain unsolved. Specifically:

combining the data collector, the recovery authority and the data analysts:

Merging the data collector with the recovery authority and the data analysts induces a security / availability problem. On one hand, one has to trust the data collector. Thus, it needs to be highly secured and handled by a restrictive access policy. On the other hand, the data collector has to be accessible for all the data analysis, requiring availability.

This conflict becomes even more apparent if the secret key k used by the data collector to choose the pseudonyms is protected by a sharing scheme. Specifically, whenever the data transfer protocol is launched, the individuals holding a key share need to meet in person, pool their shares, and make the key available to the data collector. If data transfer protocols happen rarely, say once a year, this may be feasible. Yet, it does not scale to regular data collection (say, once every month, or every week).

the data collector as big brother: The scheme provides only weak anonymity (see arguments below), but not strong anonymity, giving the data collector all power to observe individual medical histories. To see this, notice that the data collector can trivially link a given identity ID to a pseudonym P by simply computing $\tilde{P} = h(ID)$ and $P = ENC_k(\tilde{P})$. In other words, if the data collector is interested in the personal medical career of some person, it can have a look completely on its own.

Similarly, the data collector can deduce the identity from a pseudonym P on its own, even though some more effort may be involved: From P , the data collector can derive \tilde{P} and now has to find an ID such that $\tilde{P} = h(ID)$. This is easy, as the space of ID is only as large as the population of Switzerland. Given that there are only about 7 million people, launching a brute force attack over these ID s is trivial. The only tricky part is getting the 7 million personal identifying information, but given that the federal office for statistic is most probably the data collector, this data should not be too hard to get.

data analysts: When data is passed over to external analyst, the pseudonyms P are not altered. This allows several external analyst, each holding a data subset from the data collector's database, to combine their data and thereby augment their view on the individual histories. This does not affect the anonymity as long as the pseudonym is not resolved. Yet, it makes it hard to enforce that a certain data analyst may only get a certain subset of data. Refreshing pseudonyms on a yearly basis as proposed by the system addresses this problem, yet may be too coarse grained.

integrity: Integrity is not guaranteed as the data collector accepts messages without verifying any signatures, and thus, would also accept a message that was never sent by a source but by some malicious adversary that has access to the network. We remark that this could be easily fixed by using SSL/TLS connections with client authentication between the sources and the data collector. Yet, in the original scheme, one also intended to allow data being transferred by a physical device (as for example a CD-Rom). In this case, an explicit authentication code by the source is needed to guarantee authenticity of the data.

6 A Scheme Providing Strong Anonymity

In the following, we propose a solution that provides strong anonymity. Our scheme follows closely the lines of the previous scheme, yet overcomes its limitations by the following main ideas:

- In the previous scheme, the data collector not only collects the data, but also takes on the role of the recovery authority and data analysts. We factor these specific tasks out of the data collector, and assign them to the corresponding entities. This is essential to establish strong anonymity.
- The previous scheme exported data directly from the data collector's database. We use a special export protocol, which anonymizes these records for every export, thereby making it harder for two data analysts to merge their sets and augment their respective views on individual medical histories.
- The previous scheme did not authenticate sources. We base our system on a public key infrastructure that allows us to authenticate all involved entities. This way, our scheme guarantees integrity of the data.

6.1 An Informal Overview

The following is a conceptual overview of our scheme that blends out the cryptography behind it. For details on the actual implementation, we refer to the next section.

The Data Transfer Protocol. Consider a source holding a medical record $(ID, data)$. The basic idea of our data transfer protocol is as follows (see Figure 4 for an illustration):

1. The source first generates a pseudonym \tilde{P} such that *only* the recovery authority (and the source itself) can link \tilde{P} to ID . It then sends $(\tilde{P}, data)$ to the data collector.
2. When the data collector receives the message $(\tilde{P}, data)$, it asks the recovery authority for a unique pseudonym P for the patient hidden by the pseudonym \tilde{P} .
3. When the recovery authority receives such a request from the data collector, it derives the patient's identity ID defined by \tilde{P} . It then checks if it already has given out a pseudonym P for this patient. If not, it chooses a pseudonym P at random, remembers the relation (P, ID) , and sends (\tilde{P}, P) to the data collector.
4. When the data collector receives (\tilde{P}, P) , it stores $(P, data)$ in the database (5).

Since the recovery authority chooses the pseudonym P for a patient ID at random, the protocol guarantees that neither the source nor the data collector can link the pseudonym P to the identity ID (and vice versa). Furthermore, since the recovery authority chooses only one pseudonym for every identity, the protocol ensures that if two sources transfer a record for the same patient ID , then the data collector will store these records under the same pseudonym.

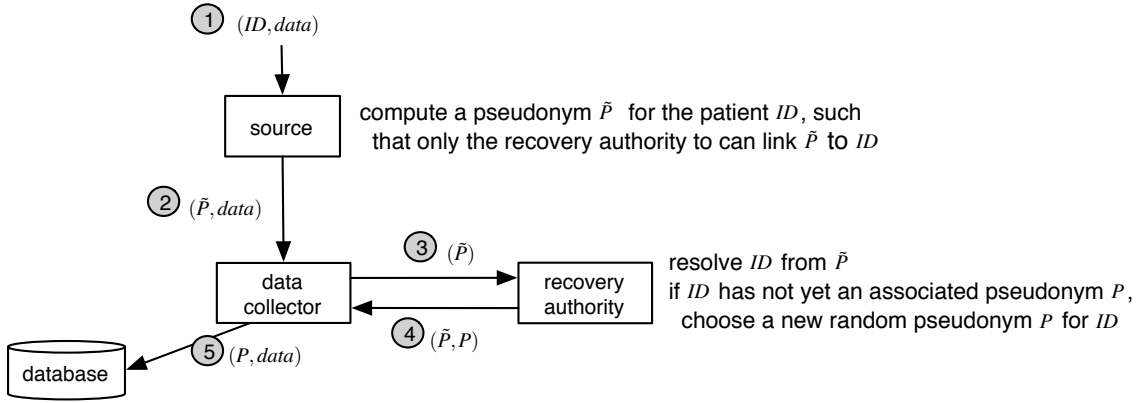


Figure 4: Data Flow in the Data Transfer Protocol

Notice that we do not address integrity in this high-level description. One can guarantee integrity by having the sources sign the payload on their data, and appending this signature to the records. See section 6.2 for details.

In practice, it may occur that a record has to be deleted. To do this, the source can use the same data transfer protocol, but mark the record being transmitted with a flag `delete`. The data collector then deletes the record under the corresponding pseudonym, rather than inserting it.

The Data Export Protocol. The basic idea of our data export protocol is as follows. For every pseudonym P that occurs in the data records to be exported, the data collector chooses a fresh pseudonym P' at random. It then replaces all pseudonyms P with their fresh counterpart P' in the records to be exported, and sends the records to the data analyst.

Apart from these records, it also sends a token to the data analyst that can later be used by the recovery authority to resolve the original pseudonym P from a fresh pseudonym P' .

Refreshing the pseudonyms for every export has the advantage that two different data analysts cannot pool their data to augment their historical data on individuals.

The Recovery Algorithm. The recovery algorithm is straight-forward: Suppose the recovery authority receives a pseudonym P' and a token from a data analyst. By the definition of this token, the authority derives the original pseudonym P as stored by the data collector. Since the recovery authority itself issued this pseudonym, it can easily derive the corresponding identity ID .

6.2 Technical Details

Our solution heavily relies on the existence of a PKI. Specifically, we assume that every source S_i , the central data collector C , the recovery authority R , and every external data analysts A_i have a private and public key-pair for encryption and signing. We will denote the encryption keys by pk_{S_i}, sk_{S_i} and the signing keys by $\bar{pk}_{S_i}, \bar{sk}_{S_i}$, where the subscripts indicate the entity owning the key.

We assume that the encryption scheme enjoys *semantic security (under adaptive chosen message attacks)*. Intuitively, this means that it is impossible to distinguish the encryption of two known messages m_1 and m_2 if the private key is unknown. RSA-OAEP [6] is an example of such a scheme. We denote the encryption of a message m under a public key pk_i by $\mathcal{E}_{pk_i}(m)$, and the decryption of a ciphertext c as $\mathcal{D}_{sk_i}(c)$.

We assume that the signature scheme is *existentially unforgeable (under adaptive chosen message attacks)*, which means that it is impossible to generate a message and signature pair without knowing the corresponding private key. We denote the signing of a message m under a private key $\tilde{s}k_i$ by $\mathcal{S}_{\tilde{s}k_i}(m)$, and the verification of a signature σ on a message m as $\mathcal{V}_{\tilde{p}k_i}(\sigma, m)$, which outputs either one for valid, or zero for invalid.

We also assume that all entities are connected to the data collector by secure links (this is easily achievable given the PKI).

Data Transfer Protocol. Let $(ID, data)$ denote the record that a source S intends to transfer to the data collector C . The source therefore encrypts ID under the public key of the recovery authority, and sends this encrypted identity \tilde{P} along with the corresponding record data to the data collector.

The data collector forwards \tilde{P} to the recovery authority, which decrypts \tilde{P} to get ID . It then encrypts ID under a symmetric key k , and sends this encryption P to the data collector. Upon receiving this message, the data collector then stores a tuple $(P, data)$ in its database.

We remark that the symmetric key k used by the recovery authority is the same for every identity ID , and is chosen at system initialization.

The protocol has to be extended slightly such that the recovery authority only provides pseudonyms for \tilde{P} 's that were produced by a source. Otherwise, the data collector could simply compute \tilde{P} for the patient he is interested in, and ask the recovery authority for the corresponding pseudonym. To prevent this, the source appends a signature to ID before it encrypt and sends it to the data collector. Furthermore, the recovery authority only provides the pseudonym if the request comes from a valid source.

Algorithm 6.2 summarizes the computational steps in more detail. We write $ENC_k(m)$ to denote a symmetric encryption of a message m under a key k .

Algorithm 1 The Data Transfer Protocol

source S :

upon input $(ID, data)$
 compute $\sigma \leftarrow \mathcal{S}_{\tilde{s}k_S}(ID)$ and $\sigma_d \leftarrow \mathcal{S}_{\tilde{s}k_S}(data)$
 compute $\tilde{P} \leftarrow \mathcal{E}_{pk_R}(ID||\sigma)$, where $||$ denotes concatenation
 send the message $(\tilde{P}, data, \sigma_d)$ to the data collector C

data collector C :

upon receiving $(\tilde{P}, data, \sigma_d)$ from source S :
 if $\mathcal{V}_{\tilde{p}k_S}(data, \sigma_d) = 0$ **then** abort
 send (\tilde{P}) to the recovery authority R
 wait for receiving an answer (\tilde{P}, P) from R
 if $P = NULL$ **then** discard the record
 else store $(P, data)$ in the database

recovery authority R :

initialization: choose a random secret key k for the symmetric encryption algorithm ENC
upon receiving (\tilde{P}) from the data collector C :
 compute $ID||\sigma \leftarrow \mathcal{D}_{sk_R}(\tilde{P})$
 if $(\mathcal{V}_{\tilde{p}k_S}(ID, \sigma) = 0)$ **then**
 send $(\tilde{P}, NULL)$ to C
 else
 compute $P \leftarrow ENC_k(ID)$
 send (\tilde{P}, P) to C

Remarks: The above protocol would put quite some load on the recovery authority if it is invoked for every data record separately. One can avoid this by processing the records in batches, which

contain all records of a certain time period (say one day, or one week). In such a batch, there may be many records for the same patient ID . If this is the case, the source may produce only a single encryption \tilde{P} per ID per batch, and re-use \tilde{P} for every record of patient ID . This way, the data collector would have to query the recovery authority only once for each patient in a source-batch.

We could also allow the source to send $(\tilde{P}, data)$ directly to the recovery authority, which then follows the protocol as described. However, we do not recommend this as it would put a high load on the recovery authority, as all the payload data goes through it, and the above batching would not be possible anymore. Furthermore, to ensure the security of the recovery authority, it makes sense to restrict access to this entity to only the data collector.

Data Export Protocol. Suppose a data collector C needs to export a set of records to a data analyst A . If the data analyst requests unlinked-data, i.e., data that does not allow to link records of the same individual to each other, then C sends all records requested directly to A with the pseudonyms P replaced by $NULL$.

If the data analyst requests linked-data, then A first chooses a random symmetric encryption key k . It then sends for every record $(P, data)$ a record $(ENC_k(P), data)$ to A . After the last record, it encrypts the key k under the public key of the recovery authority, and sends this token T to the data analyst. The token will allow later to recover the identity behind a specific pseudonym with the help of the recovery authority.

Recovery Algorithm. The recovery algorithm works as follows. The recovery authority A takes as input a de-anonymizing token T and a pseudonym P' from a data analyst. It then computes

$$k' \leftarrow \mathcal{D}_{sk_R}(T),$$

and outputs the patient's identity as

$$ID \leftarrow ENC_k^{-1}(ENC_{k'}^{-1}(P')).$$

A Remark on Supporting Multiple Identities. A data record may contain not only the patient identity, but also the identity of the treating doctor, or of the treating nurse, and so on. It may be desirable to anonymize these identities, as well. This can be done using the same techniques as we use for anonymizing the patient's identity.

Care has to be taken, however, that a person, which is both, a doctor and a patient, receives a different pseudonym for each of its roles. This can be guaranteed as follows. Recall that the recovery authority computes the pseudonym for an identity ID by encrypting it under a key k using a symmetric encryption scheme.

If we have multiple roles of identities ID_1, \dots, ID_n to be anonymized, then the recovery authority has to use different keys k_1, \dots, k_n to generate the pseudonyms, one key for each role.

6.3 Verification of Requirements

We have to show that linkability, strong anonymity, recoverability, and integrity holds as defined in Section 3.2.

linkability: Consider two records $(ID, data)$ and $(ID', data')$ that two (different) sources send to the data collector through the data transfer protocol. We have to show that if $ID' = ID$, then the data collector eventually stores two records $(P, data)$ and $(P', data')$ with $P = P'$.

When the data collector receives $\tilde{P} = \mathcal{E}_{pk_R}(ID)$ and $\tilde{P}' = \mathcal{E}_{pk_R}(ID')$, it forwards both pseudonyms \tilde{P} and \tilde{P}' to the recovery authority R . By the 'correctness' property of the encryption scheme, R will decrypt both pseudonyms to the identity ID , and thus, will send the same pseudonym P to the data collector.

strong anonymity: Consider a special scenario, where we only have one source treating only two patients ID and ID' . Suppose the two patients are mapped to pseudonyms P and P' by the recovery authority, and suppose the data collector knows the patient identities, but not the mapping to their pseudonyms. We have to show that if the source follows the protocol, and if neither of the pseudonyms is resolved by the recovery authority, then the data collector and the data analysts together can guess the correct identity with probability $1/2$.

Since the data analysts get all their data from the data collector, it suffices to show that the data collector on its own cannot guess the identity behind the pseudonyms with more than probability $1/2$.

We show this by contradiction, i.e., we show that if the above does not hold, we could use the data collector to break the semantic secure encryption scheme used to encrypt the identities. Recall that semantic security means that it is infeasible to distinguish the two random variables (m_1, m_2, c_1, pk) and (m_1, m_2, c_2, pk) , where c_1 and c_2 are the encryptions of m_1 and m_2 , respectively, under the public key pk .

Given a successful data collector (i.e., a collector that can link the pseudonyms to the identities), we build a distinguisher for the semantic encryption scheme as follows. We use the data collection with the given data collector as a sub-component as follows. We first define ID and ID' to be equal to m_1 and m_2 from the encryption scheme to be broken, and let the public key of the recovery authority be pk . We then feed the source with two records $(ID, data)$, and $(ID', data')$, and let the system run. Notice that we do not know the secret key sk of the public key that we assigned to the recovery authority. Thus, the recovery authority cannot derive the identities by decrypting the pseudonyms \tilde{P} and \tilde{P}' . However, since we control the entire system (except for the data collector), we know which pseudonym belongs to which identity without decrypting. So we can simply let the recovery authority choose the pseudonyms for each patient at random and send them to the data collector.

So far, the data collector cannot tell a difference between this 'simulated' environment and the real environment, as the distribution of all the messages (in particular, the pseudonyms P and P') are identical. This does not change, either, if the data collector tries to query the recovery authority on any self-computed encryption of an identity ID , since by the security of the signature scheme used by the source, the data collector cannot compute a signature on ID which would be required by the protocol.

This implies, together with the assumption that C is successful in the real environment, it will also be able to predict which pseudonym corresponds to which patient with more than probability $1/2$, i.e., it can distinguish (ID, ID', P, pk) from (ID, ID', P', pk) with non-significant probability. We can use this output directly to break the semantic security of the encryption scheme.

A similar argument can be given to show that also the source cannot link a pseudonym P to one of its patients ID .

recoverability: Follows directly by the protocol.

integrity: We have to show that if the data collector receives a record $(P, data)$ from the data transfer protocol, then the recovery authority maps P to some identity ID such that there exists a source which provided the record $(ID, data)$ as input to the protocol.

We start with the fact that the data collector received a record $(P, data)$. By the protocol, it must have asked the recovery authority for a pseudonym given some \tilde{P} . Thus, there exists some ID such that $P = ENC_k(ID)$, where k is the key used by the recovery authority. This implies the first part of integrity.

To see the second part, note that if the data collector receives as output a record $(P, data)$ from the data transfer protocol, we may assume that it received a record $(ID, data, \sigma_d)$ from some source S , where σ_d is a valid signature on $data$ under the public key pk_s . It follows by

the unforgeability of the signature scheme that source S has received the record as input to the protocol.

6.4 Offline Recovery Authority

Notice that in the proposed scheme, the recovery authority has to be online during the data transfer protocol. On the other hand, it needs to be very secure. Since these two requirements conflict to some extent, it would be desirable to have a data transfer protocol where the recovery authority does not have to be involved and can stay offline. In such a system, the authority could live in a safe-like room, completely disconnected from any network.

Despite these advantages, an offline scheme also has some security drawbacks: If some source and the central data collector behave malicious, they receive essentially the power of the recovery authority. This is because they can compute—by definition of an offline scheme—the pseudonym of an arbitrary patient without interacting with any other entities. Since there are only a limited (and well-known) set of patients, they can easily compute a complete mapping between identities and pseudonyms.

We remark that theoretically, this is also possible in any online scheme. This is because one cannot verify if a data record provided by a source represents a treatment that indeed happened at that source. Thus, a malicious source can simply invent a data record for every patient, insert it into the system, and compare it with the output of the data collector to build a complete mapping between identities and pseudonyms (see Section 3.3 for details).

In this attack, however, the recovery authority is involved for every mapping that is computed. Thus, in practice it should be feasible to detect such an attack by heuristics on the recovery authority side, except if many sources are malicious. Furthermore, the introduction of a medical card will allow to have a patient confirm the integrity of a data record, and thus prevent such attacks completely.

Notice that this drawback heavily relies on the fact that one entity receives exactly one pseudonym. If this linkability is not required, the above arguments do not hold anymore, and an offline scheme is certainly the desired approach. Yet, if linkability is required, we recommend to use an online scheme as the one proposed in the previous section.

7 The Strong Scheme in Practice

In this section we present a possible implementation of the scheme providing strong anonymity (section 6). This implementation has several requirements concerning system architecture and performance. First, we give an overview over the services their interactions and locations. Then we present an estimation of hardware requirements for these services.

7.1 System Architecture

Three main areas have to be distinguished:

- A database area, where the main database is located together with access-controlling systems.
- A recovery area, which must be located outside the database area for security reasons.
- The Internet, from where requests for access to the database area origin.

Sources connect over the Internet to the database area to submit their data. They have to pass a reverse-proxy where all requests must be authorized and authenticated before they are passed on. Data-analysts using an export client can only query the database by connecting over the

reverse-proxy, where they are also authenticated. This allows to build only one entry-point to the database area and have all other connections in its own network, providing higher security than having several entry-points.

The proxy server passes all requests either to a server dedicated to import data from sources or to a server dedicated to query the database. These servers have only the minimal needed functionality for their job: The import server has only write-access to the database (except read-access on some log-tables to ensure no data is committed twice), while the export server has only read-access. This modularity allows to extend the performance of the system by adding additional export or import servers and having the reverse-proxy act as a load-balancer.

The import server must be able to communicate with the recovery to receive the pseudonyms P_{id} . We propose to connect the import server directly with the recovery over a dedicated leased line, protected on both side with a firewall.

Figure 5 gives an overview over the proposed servers. All data transfers should be handled over SSL/TLS connections with server and client authentication. If the load on the import and export servers are low, they could be merged into one machine.

The data has not only to be secured to unauthorized access, but also to erroneous manipulation and hardware failure. We propose thus to use database-replication tools to have always an up-to-date copy available. Additionally, it would be useful to create backups regularly to a removable device for creating remotely storable backups. This could be done using tape decks or a similar technology. The records should be further protected by using an encrypting hard disk or file system for the database storage.

The database can get quite large (see “Required Performance” below). To store the data, a RAID-System (Redundant Array of Inexpensive (or Independent) Disks) seems inevitable. We propose to use a RAID 0+1 or a RAID 5 system to create a large and failure-tolerant disk-set.

Note that this concept does not ensure 100% up-time, but it protects quite effective from data loss. Because the system is mainly used as a data collecting service and (indirectly) for data analysis, it is not critical if the service is not available for a day or two. But it would be much worse if the data of the last five years would be lost due to a hard-disk crash or a similar incident.

The different sources use many different internal systems to manage their data. It is far beyond the scope of this project to unify all those systems. Thus the sources must be able to generate correctly formatted records from there systems on their own to submit them to the central data collector. For security reasons and reasons of manageability, the sources are given a client software which implements the protocol to submit the records to the data collector. The client offers one or more ways to load the records, e.g. reading flat files (CSV or similar), selecting the data from a database-table over ODBC, or a similar standards-based interface. This client application marks the outermost part of the system that tangents the sources.

7.2 Performance Estimations

Collecting the data from all hospitals, health insurance funds and care providers generates quite an amount of data. Assuming we have every year 5 million patients, each being treated for 6 diseases. Every treatment of a disease generates 100 records in average. This means that every year 3 billion records have to be transmitted to and stored in the database. To simplify the following calculations and to allow for a potential future increase in records we will use the number of 5 billion records per year (about 13,7 million records per day). This already shows that the data has to be transfered regularly and not only once a year as in [1].

We assume that an average record has a size of 1 KB. This may sound huge in the first moment, because in every record only a few values are submitted. But as soon as one or more values are encrypted in the records, they require much more place. Storing all those records requires thus about 5 TB for each year (about 14 GB for each day). This can be easily handled by todays RAID-systems or networked storage.

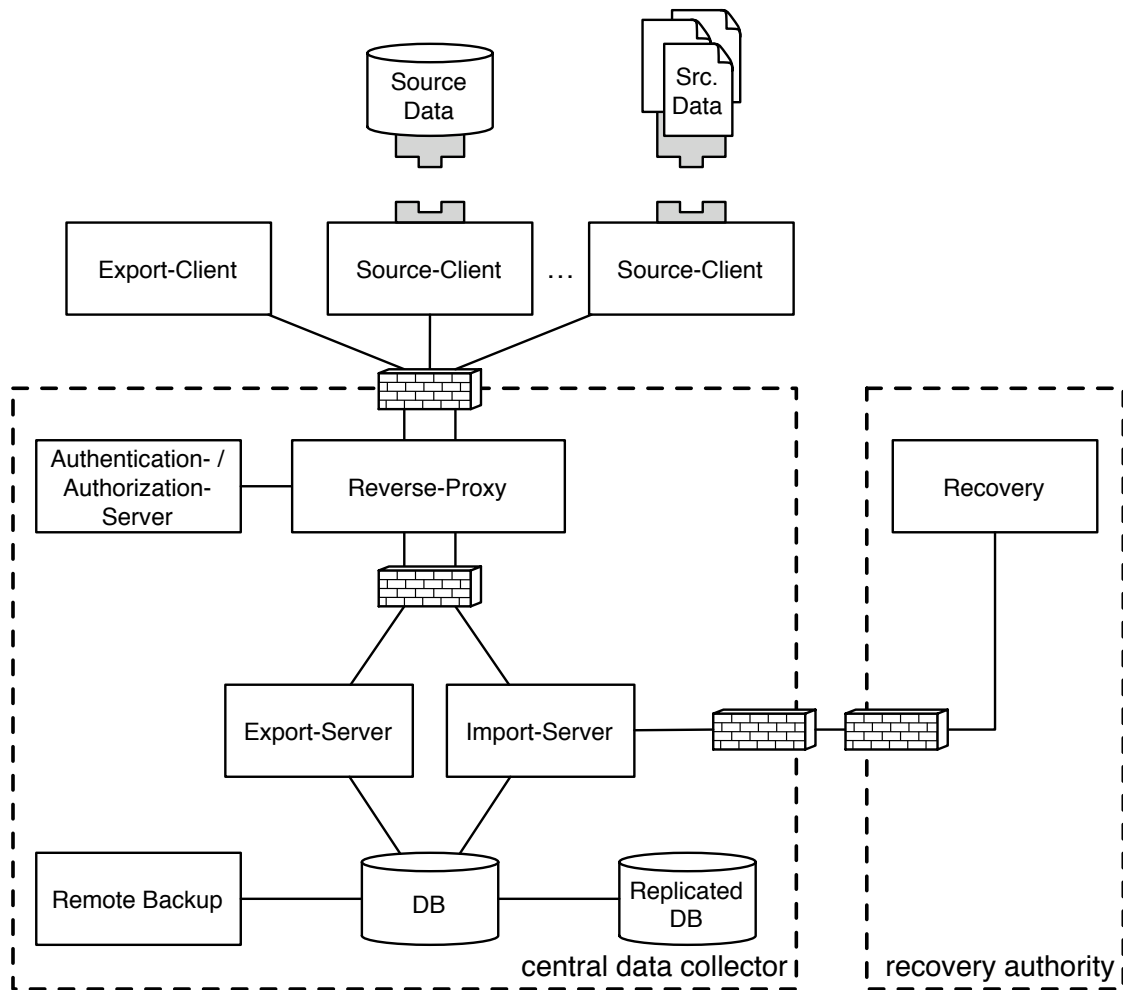


Figure 5: Proposed Hardware Strategy

All these records have to be transferred over the Internet. The 14 GB per day correspond to a steady flow of about 160 KB per second, or 1.3 Mbit/s. Assuming a peak of 15 to 20 times the throughput, the data collector must be connected using a 20 Mbit lane. Assuming that about 100 sources would deliver the data, the load for each could easily be handled by the upstream of today's broadband connections.

As the aforementioned numbers show is the physical transmission no problem. But before and after transmission, in every record some data must be sign and encrypted, or decrypted and the signature verified. Table 1 shows for each entity how many cryptological functions must be applied per record, while table 2 shows the number of cryptological functions applied for each entity per day.

We can then calculate the time a server is busy to perform the cryptological functions by making an assumption about the number of functions a modern server can perform today (see table 3). The times are shown on table 4. The final daily loads are summed up in table 5. As can be seen, the recovery is the most heavily used component. Note that these numbers are only rough estimations, and that no hardware acceleration or optimizations were considered. A possible optimization for the recovery is to cache the calculated pseudonyms. Processing over 13 million records on a day while there are only 5 million patients, the load can be cut at least in half in theory. In practice, the factor will be even larger because it is unlikely that all 5 million patients are treated in the same day or week the data consists of.

	sign	verify signature	asymmetric encryption	asymmetric decryption	symmetric encryption and decryption	SSL/TLS (1KB-Block)
source	1		1			1
collector		0,001				3
database						1
recovery		1		1	1	2

Table 1: Number of cryptological functions applied per data record

	sign	verify signature	asymmetric encryption	asymmetric decryption	symmetric encryption and decryption	SSL/TLS (1KB-Block)
source	137k		137k			137k
collector		13,7k				31,1m
database						13,7m
recovery		13,7m		13,7m	13,7m	27,4m

Table 2: Number of cryptological functions applied by each entity per day

	sign	verify signature	asymmetric encryption	asymmetric decryption	symmetric encryption and decryption	SSL/TLS (1KB-Block)
number of functions per second	500	10 000	10 000	500	250 000	20 000

Table 3: Assumed performance of a modern single CPU server (2005)

	sign	verify signature	asymmetric encryption	asymmetric decryption	symmetric encryption and decryption	SSL/TLS (1KB-Block)
source	274		14			7
collector		1,3				1 555
database						69
recovery		14 000		27400	55	1 370

Table 4: Assumed time in seconds to handle daily load for each function

	total time in seconds	load over 24 hours
source	295	0,34%
collector	1 556	1,8%
database	69	0,08%
recovery	41 825	48%

Table 5: Average Processor Loads

8 Further Applications and Extensions

The privacy-preserving data collection offers interesting applications for example for payment, controlling and risk balancing. In combination with a medical insurance card even anonymous treatment or access to an online medical data collections would be possible.

8.1 Medical Insurance Card

On the 22nd of June 2005, the federal council passed a concept for the initiation of a swiss medical insurance card. This concept comprises that from 2008 every insurant has to present his card for drawing any medical benefits. We highly recommend to provide such medical insurance cards with a chip having cryptographic capabilities (as for example a smart card [4, 7]). This would open many very interesting possibilities.

- **Unique Identification:** A medical card is the ideal medium for saving a unique identifier as described in section 4. It can even be used to generate a per treatment unique pseudonym which will be used for the privacy-preserving data collection.
- **Anonymous Treatment:** A medical insurance card does not need any imprints of personal data. So, a patient could take his card and go to a care provider and ask for a treatment. Based on a zero-knowledge proof [8] the care provider verifies that the patient has a valid insurance. After verification, the care provider only receives pseudonym which was automatically created by the card. After the treatment, the care provider sends the bill to a central billing agency. This agency debits the costs to the accurate insurance and forwards the detailed data to the privacy-preserving data collection.
- **Online Medical Data:** A medical card could be used for authorization granting dedicated access to an online database containing detailed medical data about its owner as for example x-ray pictures, blood group, and prescribed medicaments.

8.2 Payment and Controlling

Payment of care providers is still based on two old models.

1. **Tier Payant:** The care provider directly sends the bill to the accurate insurance. The patient never sees a bill and has no possibility of control.
2. **Tier Garant:** The care provider sends the bill to the patient. The patient pays the bill and sends it to the insurance for refunding.

The privacy-preserving data collection could be used as central billing agency.

8.3 Balance of Risk

A very hot topic not yet mentioned is the balance of risk between the insurance carriers. Right now there is an ongoing political and juristic discussion on how to justly revise this balance. In this discussion, the privacy-preserving data collection offers very interesting possibilities. Its detailed data can be used for risk calculation and balancing between the insurance carriers.

9 Discussion and Conclusion

In this work, we investigated the problem of a privacy-preserving data collection, and proposed a system that offers a reasonable protection for the privacy of the individuals whose data is collected.

A distinguishing property of our system is that it has a designated entity, the recovery authority, which is responsible for the anonymity of the individuals. Specifically, even the central data collector cannot recover the identity behind a pseudonym.

This contrasts with the previous system proposed in [1], where anonymity was established and guaranteed by the central data collector, and no recovery authority existed.

We believe that having to trust the central data collector is a major drawback of this system. This is because the data collector is likely to be a complex system, given the number of clients it communicates with, and the amount of data it collects. In practice, such a system is harder to secure than a recovery authority, which has a very specific task and can be designed in a simple and effective way.

Another difference to the previous system is that we factor data analysis out of the data collector, i.e., rather than allowing data analysis directly on the data collector's data, we suggest to use an export protocol which re-randomizes the data collector's pseudonyms before the data is given to an analyst.

This ensures that two data analysts cannot pool their data and thereby augment their view on the individuals, as the same individual will have different un-linkable pseudonyms for every data analyst.

And finally, we point out the importance of a mechanism to verify that a certain data record being collected is genuine, i.e., that the record represents an event that actually happened in the real world. Specifically, without such a mechanism, anonymity cannot be ensured with respect to malicious data providers (sources).

At the end of this report, let us quickly glance at Denmark. They already have successfully implemented an overall medical database (Medcom) [3] in the late eighties. Medcom offers open, unanonymized access for care providers and statisticians as well as, on their data restricted, access for patients. Such a model is unthinkable for Switzerland with its strict data protection law and its background, especially due to the 'Secret Files Affair'. But the example of Denmark should encourage us to realize a privacy-preserving data collection which fulfills all necessary requirements.

References

- [1] Bundesamt für Statistik. Der Datenschutz in der Medizinischen Statistik - Statistik der stationären Betriebe des Gesundheitswesens, 1997.
- [2] Bundesamt für Statistik. Gesundheitsversorgung, Kosten, Finanzierung - Kennzahlen, 2005.
- [3] Markus Hofmann. NZZ Folio: Spitzenposition für Dänemark, September 2005.
- [4] International Telecommunication Union (ITU). Public-key infrastructure (x.509).
- [5] David-Oliver Jaquet-Chiffelle. Description des méthodes envisagées pour protéger la confidentialité des données personnelles. 1997.
- [6] RSA Laboratories. PKCS #1: RSA Cryptography Standard, 2003.
- [7] RSA Laboratories. Public-key cryptography standards #11 and #15, 2003.
- [8] S. Micali S. Goldwasser and C. Racko. The knowledge complexity of interactive proof-systems. *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, 1985.
- [9] J. L. Massey X. Lai. A proposal for a new block encryption standard. pages 389–404. Springer-Verlag, 1991.