Towards Secure Cloud Computing through a Separation of Duties

Christian Henrich, Matthias Huber, Carmen Kempka, and Jörn Müller-Quade
surname.name@kit.edu

Abstract: Cloud Computing offers many opportunities but also introduces new risks. A user outsourcing his database into the cloud loses control over this data. While the service provider often secures the data against external threats using standard techniques, the service providers themselves have to be trusted to ensure privacy. This work proposes a novel approach to provide security for database services without the need to trust the provider. We suggest employing a *separation of duties* by distributing critical information and services between two or more providers in a way that the confidentiality of a database can only be compromised if all providers are corrupted and work together. We also present a formal security notion for such a database.

1 Introduction

Cloud Computing is "a model for enabling convenient, on-demand network access to a shared pool of computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [NIS09]

Inherent to this model are privacy problems. By using services in the cloud clients lose control over their data. Current security mechanisms focus on protecting the data transfer to and from the service provider. But the threat of insider attacks keeps many potential customers from using cloud computing for critical applications.

For a storage service, providing protection against insider attacks can be achieved easily by encrypting all data. But this prevents the server from performing any meaningful operation on the data. Services more complex than simple data storage require advanced techniques.

There are cryptographic methods [GMW87] that in principle can solve many privacy problems, especially since a fully homomorphic encryption [Gen09] was discovered in 2009. Due to high costs, however, these methods are infeasible and cancel the benefits of outsourcing services. Nevertheless we need privacy and security guarantees for Cloud Computing so it can also be used in sensitive scenarios.

This paper is organized as follows. In the remainder of this section, we discuss related work. We apply our concept separation of duties to a database service in Section 2. In Section 4, we present our new security notion for anonymization procedures. Finally we provide a proof sketch that our database service fulfill this notion in Section 4. Section 5 summarizes our results and states open problems.

1.1 Related Work

There are cryptographic solutions for two or more parties cooperatively computing a certain function over a set of data without any party learning anything about the input of the other parties. Using an interactive protocol these secure multiparty computations

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4

[GMW87] thus solve many privacy problems. The only problem is that for each party the computation cost is higher than computing the whole function on the complete input without any other party. This makes the concept of multiparty computation for outsourcing services unsuitable and in fact pointless if the client is the only one with private input.

For Databases, thare are many practical approaches in the fields of Secure Database Outsourcing and Privacy Preserving Database Disclosure, which will be revied in the remainder of this section.

1.1.1 Secure Database Outsourcing

In 2002, the concept of *Database as a Service* emerged. Consequently much work has been done considering the privacy of outsourced databases. We identified two classes of approaches, that potentially suppport searching a database in sublinear time: The *coarse indices approach* [HILM02] and the *distribution approach* [ABG$^+$].

All approaches try to find a tradeoff between security and efficient support for as many types of query as possible. The coarse indices approach suggests to encrypt a database tuple-wise and to create coarse indices. This enables efficient execution of exact-match and range-queries. This scheme, however, does not support efficient execution of substring-queries. The distribution approach proposes to distribute a database to different servers in order to meet so-called *privacy constraints*. If a privacy constraint cannot be met by distribution, they propose to use encryption. This scheme supports the efficient execution of queries to attributes whose values are not encrypted. Queries containing encrypted attribute values, however, are not supported efficiently. Besides the approaches presented above there are approaches that rely on special hardware [KC04] or assumptions about the input data [BBA07].

1.1.2 Privacy Preserving Database Disclosure

In privacy preserving database disclosure tries to preserve the privacy of individuals represented in a disclosed database. The anonymity properties *k-anonymity* [SS98], *l-diversity* [MKGV07], and *t-closeness* [LL07] describe requirements a released database has to fulfill. The idea behind these properties is to generalize the database before disclosing it in order to make it impossible to map an individual to a single database entry. It is well known that these properties have weaknesses. They are vulnerable to attacks based on background knowledge (e.g. *homogeneity attack*, *skewness attack*, and *composition attack*).

1.2 Our Contribution

We introduce a new security notion that can be applied to outsourced databases. This notion is different to the notions defined in [BBA07] and in [Dwo06] since our notion does not require assuptions about potential input data. We apply separation of duties, a concept introduced in [Hub10] an in [HHK$^+$10] to a database service and show this service fullfills this notion. Hereby we only consider static analysis.

2 A Secure Database-Service through Separation of Duties

Separation of Duties suggests to partition a service in order to improve its security. In this section, we apply separation of duties to a database service. In Section 4, we

ISBN 978-3-88579-286-4

argue that this service fullfills the security notion we will define in Section 4. Consider a customer relationship database like the one in Figure 1.

id	name	surname	bank account
1	Alice	Smith	573535635
2	Bob	Smith	436343346
3	Alice	Jones	343462624

Figure 1: An example database.

We want to search for a name and get the corresponding bank account number but will never search for a bank account number. For this database, we propose to separate the indices from the database, and apply encryption (cf. Figures 2 and 3). We use a probabilistic encryption *Enc*. We apply *Enc* to each tuple in the data table an to the pointer entries in the index tables.

keyword	pointer
Alice	*Enc*(1, 3)
Bob	*Enc*(2)

(a)

keyword	pointer
Smith	*Enc*(1, 2)
Jones	*Enc*(3)

(b)

Figure 2: Separated indices of the database with encrypted pointers.

id	E(name,surname,bank account)
1	*Enc*(Alice,Smith,573535635)
2	*Enc*(Bob,Smith,436343346)
3	*Enc*(Alice,Jones,343462624)

Figure 3: The encrypted data table

The system depicted in Figure 4 shows how the resulting tables are distributed. The tables of Figure 2 are placed on the servers "index_1" and "index_2" respectively. The table in Figure 3 is placed on the server "data". Because of this separation, before retrieving data from or writing data to the data table, the client has to query the indexing server(s) for pointers, or create new pointers. Note that, despite the communication and encryption overhead, search time is still sublinear.
Also note that this example is not secure in the strict sense of classical cryptography as information about the structure of the data, e.g., the number of different attribute values, may leak.

3 Security of Anonymization Functions

Though k-anonymity, l-diversity and t-closeness are a good first indication for anonymity, they cannot directly be used as a notion of security of services. Consider a storage service where the database is anonymized by probabilistically encrypting

ISBN 978-3-88579-286-4

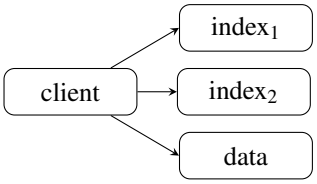

Figure 4: Schematic of our proposed separation of our example (CRM database).

each entry. The resulting database clearly is not k-anonymous, as each combination of ciphertexts will be unique with overwhelming probability. However, probabilistically encrypting a database should be considered secure by any reasonable notion of security.

Even worse, the anonymization process may contain a subliminal channel leaking information without violating the k-anonymity of the resulting database.

Consider the following toy-example: Imagine an anonymization procedure for a database which completely deletes the column with the quasi identifier *street name* whenever a rich VIP is contained in the database. Such an anonymization procedure seems to especially protect rich and famous people, however, the combination of *rich* and *famous* could hold for less than k persons. Still one can derive this property, holding for less than k entries of the database, from the fact that the street names were completely deleted. So even if the result of an anonymization procedure is k-anonymous the "anonymized" database could still leak sensitive data. This shows that we need to define anonymity not for the database that results from anonymization, but for the process of anonymization itself.

3.1 A new security notion: Ind-ICP

In this section, we describe the properties of an anonymization function f that takes a database of a certain format as input and computes an "anonymized" database. Since any number of tables can be represented with the universal relation , for our purposes, we define a database as follows:

Definition 1 *Database*
A database *is a table containing arbitrary many columns We call the set of all databases DB.*

Definition 2 *Database Function*
A database function *is a function $g : DB \to DB$. We call $\mathscr{F}$ the set of all database functions.*

Examples for database functions are projection π and selection σ from relational algebra and our anonymization functions. Another special case of database functions we will use are special permutations defined as follows:

Definition 3 Π
Let $\Pi \subset \mathscr{F}$ be the set of database functions $\mathfrak{p} : DB \to DB$ such that each $\mathfrak{p} \in \Pi$ independently permutes the entries within each column of a database but leaves the information column untouched.

Intuitively, we say that f is a good anonymization if an adversary is unable to distinguish between an anonymization of the original database and an anonymization of

a version of the original database where the entries in each column have been permuted independently from each other and hence relations between entries have been eliminated.

Definition 4 *Experiment Ind-ICP$^i_{\mathscr{A}}(d)$*
Let $d \in DB$ be a database, $f \in \mathscr{F}$ a database function, $\mathfrak{p} \in \Pi$ a permutation, $\mathscr{A}$ an adversary and $i \in \{0,1\}$. Depending on i, we define our two experiments as follows:
$d_0 := f(d)$
$d_1 := f(\mathfrak{p}(d))$
$b := \mathscr{A}(d_i)$
return b

Definition 5 *Advantage of Adversary A*
For each row r_i in d exists a set M_i of k rows in d (called the k-bucket of row r_i) such that $r_i \in M_i$ and for each $\mathfrak{p} \in \Pi$ that affects only rows in M_i and leaves the entries of all other rows unchanged the advantage of the adversary $\mathscr{A}$ is

$$\mathrm{Adv}^{\textit{Ind-ICP}}_{\mathscr{A}}(d) := Max_{\mathfrak{p}\in\Pi}\left|Pr[\textit{Ind-ICP}^0_{\mathscr{A}}(d) = 0] - Pr[\textit{Ind-ICP}^1_{\mathscr{A}}(d) = 0]\right|$$

If the advantage of an adversary is smaller than ε, k-Indistinguishability under Independent Column Permutation holds:

Definition 6 *(k-Ind-ICP) with ε-Advantage*
For a database function f k-Indistinguishability under Independent Column Permutation *(k-Ind-ICP) holds iff for each database $d \in DB$, for each polynomially restricted adversary $\mathscr{A}$ the following holds:*

$$\mathrm{Adv}^{\textit{Ind-ICP}}_{\mathscr{A}}(d) < \varepsilon$$

If k-Ind-ICP holds for an anonymization function f, a database anonymized with f will at most leak information about individual entries in the original database. It does not contain information about relations between entries that would be eliminated by independently permuting the entries of each column. Note that d is chosen prior to $\mathscr{A}$.

4 Proving the k-Ind-ICP Property

In this section, we provide a sketch for a proof that the scheme described in Section 2 adheres k-Ind-ICP, where k is the number of rows in the database. We show that an adversary cannot distiguish between an transformation of a table d or the transformation of a table $\mathfrak{p}(d)$ where $\mathfrak{p}$ is an arbitrary independent colum permutation (c.f. Section).

Consider the tables depicted in Figure 5. The table 5 (a) is an arbitrary table with attributes a_i and values $v_{i,j}$.

The transformmation of both tables according to the scheme described in Section 2 yields the index tables depicted in Figure 6 and the data tables depicted in Figure 7. For the sake of simplicity, we assume that each value of table 5(a) is unique.

Let us first consider the tables in Figure 6. The keyword columns of table 6(a) an table table 6(b) are identical, and the values of the pointer column are encrypted. Assuming they thave the same length, wich can be acieved with padding, the two tables are indistiguishable with respect to the underlying encryption algorithm.

For the same reasons, the tables 7(a) and 7(b) are indistiguishable: The first colums are identical and, assuming proper padding, the values in the second colums are indistiguishable.

Consequently, the transformation described in Section 2 adheres k-Ind-ICP, where k is the number of rows in the database.

a_1	a_2	...
$v_{1,1}$	$v_{2,1}$	...
$v_{1,2}$	$v_{2,2}$	...
...	...	...

(a)

a_1	a_2	...
$v_{1,\mathfrak{p}_1(1)}$	$v_{2,\mathfrak{p}_2(1)}$	...
$v_{1,\mathfrak{p}_1(2)}$	$v_{2,\mathfrak{p}_2(2)}$	...
...	...	...

(b)

Figure 5: (a) A table with attributes a_i and values $v_{i,j}$ and (b) the same table after applying an independent column permutation

keyword	pointer
$v_{i,1}$	$Enc(1)$
$v_{i,2}$	$Enc(2)$
...	...

(a)

keyword	pointer
$v_{i,1}$	$Enc(\mathfrak{p}_i(1))$
$v_{i,2}$	$Enc(\mathfrak{p}_i(2))$
...	...

(b)

Figure 6: The index tables for attribute a_i derived from the tables depicted in Figure 5

5 Conclusion and Future Work

In this work, we applied separations of duties to a database. We also introduced a security notion, and showed that the proposed database service fulfills this notion. For future work we want to further investigate the security and the performance of our database scheme. We also want to investigate different privacy properties and security notions for databases. This leads to a better understanding of security of databases and may yield additional methods for securing database services.

6 Acknowledgments

The work presented in this paper was performed in the context of the Software-Cluster project EMERGENT (www.software-cluster.org). It was partially funded by the German Federal Ministry of Education and Research (BMBF) under grant no. ”01IC10S01”. The authors assume responsibility for the content.

id	values
1	$Enc(v_{1,1}, v_{2,1})$
2	$Enc(v_{1,2}, v_{2,2})$
...	...

(a)

id	values
1	$Enc(v_{1,\mathfrak{p}_1(1)}, v_{2,\mathfrak{p}_2(1)})$
2	$Enc(v_{1,\mathfrak{p}_1(2)}, v_{2,\mathfrak{p}_2(2)})$
...	...

(b)

Figure 7: The relation tables derived from tables depicted in Figure 5

References

[ABG+] G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, R. Motwani, U. Srivastava, D. Thomas und Y. Xu. Two Can Keep a Secret: A Distributed Architecture for Secure Database Services. *CIDR 2005*.

[BBA07] M. Bellare, A. Boldyreva und A.O'Neill. Deterministic and Efficiently Searchable Encryption. In *CRYPTO*, Seiten 535–552, 2007.

[Dwo06] C. Dwork. Differential Privacy. *Automata, Languages and Programming*, Seiten 1–12, 2006.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, Seiten 169–178, New York, NY, USA, 2009. ACM.

[GMW87] Oded Goldreich, Silvio Micali und Avi Wigderson. How to play ANY mental game. In *STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of computing*, Seiten 218–229, New York, NY, USA, 1987. ACM.

[HHK+10] C. Henrich, M. Huber, C. Kempka, J. Mueller-Quade und R. Reussner. Technical Report: Secure Cloud Computing through a Separation of Duties. `https://sdqweb.ipd.kit.edu/huber/reports/sod/technical_report_sod.pdf`, 2010.

[HILM02] H. Hacigümüs, B. Iyer, C. Li und S. Mehrotra. Executing SQL over encrypted data in the database-service-provider model. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, Seiten 216–227. ACM, 2002.

[Hub10] Matthias Huber. Towards Secure Services in an Untrusted Environment. In *Proceedings of WCOP 2010*, Jgg. 2010-14 of *Interne Berichte*, Seiten 39–46, Karlsruhe, Germany, 2010. KIT, Faculty of Informatics.

[KC04] Murat Kantarcioglu und Chris Clifton. Security Issues in Querying Encrypted Data. Bericht, 2004.

[LL07] Ninghui Li und Tiancheng Li. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. 2007.

[MKGV07] A. Machanavajjhala, D. Kifer, J. Gehrke und M. Venkitasubramaniam. l-Diversity: Privacy beyound k-Anonymity. *Cornell University*, Seite 52, 2007.

[NIS09] NIST. NIST - Cloud Computing. `http://csrc.nist.gov/groups/SNS/cloud-computing/`, 2009.

[SS98] Pierangela Samarati und Latanya Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). Seite 188, 1998.