Dynamic anonymous index for confidential data

Guillermo Navarro-Arribas[1], Daniel Abril[2], and Vicenç Torra[2]

[1] Dep. Enginyeria de la Informació i de les Comunicacions (DEIC),
Universitat Autònoma de Barcelona (UAB).
[2] Institut d'Investigació en Intelligència Artificial (IIIA),
Consejo Superior de Investigaciones Científicas (CSIC).

Abstract. In this paper we introduce a k-anonymous vector space model, which can be used as an index of a set of confidential documents. This model allows to index, for example, encrypted data. New documents can be added or removed while maintaining the k-anonymity property of the vector space.

1 Introduction

We tackle the problem of storing confidential documents in a cloud computing scenario, where the server storing the documents is not the owner of the documents. In such cases, users might like to protect their documents by, for example, encrypting them. This will ensure that the document remains confidential in the server. It is not only protected from other users, but also ensures that any intrusion performed in the systems or security breach will not lead to the disclosure of the contents of the document.

At the same time, in some scenarios it will be desirable to provide some form of index or metadata about the stored documents. The metadata could be used to perform queries on a given set of documents, but also for other information retrieval tasks. Ideally these metadata should preserve to some extent the privacy provided by encrypting the documents.

Instead of relying on a strictly cryptographic solution, in this paper we explore a novel approach inspired by the application of SDC (*Statistical Disclosure Control*) and PPDM (*Privacy Preserving Data Mining*) techniques to information retrieval. Our approach is to provide public metadata about the encrypted documents. The metadata must ensure that a minimum degree of privacy and anonymity is provided about the documents while presenting some useful information about them. We have chosen to represent these metadata as a document vector space model (VSM) [8], which is normally used in information retrieval systems. In a vector space model, it is common to represent a document as a vector of terms with an associated frequency-based weight.

In Section 2 we introduce the motivating scenario. Section 3 introduces the k-anonymous VSM, and its protection is described in Section 4. Finally, Section 5 provides some evaluation results and Section 6 concludes the paper.

2 Scenario description

We contextualize our proposal in a typical cloud computing system or more precisely as a cloud storage service. We consider a repository of documents, where each document belongs to a different user. Examples could be a repository of electronic patient health record, papers submitted to a conference, or a repository of research project proposals.

We do not deal here with the concrete protocols and processes to implement the interaction with the user. Intuitively, the user submits an encrypted document together with the document vector representing the document. The user (or client software) is free to apply any desired pre-anonymization to the document vector. Then the server adds the document vector to the VSM. This process requires, as we will see, the anonymization of such vector. The user can, on request, delete his document from the server, and its corresponding document vector is deleted from the VSM.

It is important to note that we consider the server as trusted. That is, the users trust it to correctly perform the anonymization process. Once a document vector is anonymized to be included in the VSM, it is deleted by the server. The fact that the server does not keep the original vectors ensures an additional level of security in case of an intrusion.

The main objective of our proposal is to anonymize the VSM. Compared to typical datasets used in SDC and PPDM, the VSM as presented in this work can be considered a *dynamic* dataset. We consider that there can be discretionary insertions and deletions of documents in the server, which has the implications that document vectors can be added to or removed from the VSM. There are some proposals dealing with stream data [2,4,6,7,9,5], which only contemplate insertions in the dataset. Moreover streaming data assumes that new records will be inserted in a timely basis, allowing the buffering of new records to be inserted. The concrete case of dynamic data has scarcely been treated in [16, 15]. We depart from these works to introduce the dynamic anonymization using microaggregation, and it application to VSM.

3 k-anonymous VSM

In order to maintain the document based anonymity of the VSM, we introduce the idea of k-anonymous VSM as analogous to classical k-anonymity with respect to quasi-identifiers in SDC [12,14]. We will introduce the k-anonymous VSM together with some notation. Given a set of documents D we denote as $\boldsymbol{V}(d_i)$ the vector for document d_i, such that $\boldsymbol{V}(d_i) = (w_{1,i}, \dots, w_{M,i})$, where $w_{j,i}$ is the weight associated to term j in document i. The set of document vectors $\mathcal{D} = \{\boldsymbol{V}(d_1), \dots, \boldsymbol{V}(d_n)\}$ forms the *vector space model (VSM).* The weight is normally taken to be a frequency based metric associated with the term [8].

Definition 1 *A VSM $\mathcal{D}$ is a k-*anonymous VSM *if and only if for every document vector $\boldsymbol{V}(d_i)$ in $\mathcal{D}$ there exists at least $(k-1)$ other document vectors in $\mathcal{D}$ that are indistinguishable from $\boldsymbol{V}(d_i)$.*

In this paper we deal with dynamic data, and its protection if it is not done properly can lead to inference.

3.1 Inference on multiple anonymizations of dynamic data

Several works show inference attacks on data with multiple anonymizations, including streaming data, in terms of l-diversity [2, 16]. We do not deal with confidential attributes, so l-diversity does not apply.

Even so, our VSM data can be vulnerable to inference through intersection of equivalence classes. This is a common problem in clustering based anonymization when multiple anonymizations of the same data are released [13, 9]. An attacker can reduce the cardinality for some given quasi-identifier values by intersecting different equivalence classes that contain some common records, thus, breaking k-anonymity.

In [13] the authors show that intersection can easily happen in the context of generalization. The same can be applied to microaggregation. Consider for instance the generic example from Table 1. For simplicity we will consider a single numeric attribute in the dataset (age). Note also that the record identifier r_i is given only as a reference to help understand the example, the actual anonymized dataset (VSM in this case) has only quasi-identifiers, and no other identifiers is given.

Record	Age
r_1	12
r_2	12
r_3	21
r_4	30
r_5	30

(a) Original table T

Record	Age
r_1	15
r_2	15
r_3	15
r_4	30
r_5	30

(b) Microaggregated table T_1

Record	Age
r_1	12
r_2	12
r_3	27
r_4	27
r_5	27

(c) Microaggregated table T_2

Table 1: Example of intersection in microaggregated tables.

We have an original dataset T with two different 2-anonymous microaggregated versions T_1, and T_2. An attacker with knowledge of both tables, and knowledge of the aggregation applied (in this case the arithmetic mean) can easily infer information about the record r_3. Given the aggregation operator $\mathbb{C}$, if $\mathbb{C}(r_1, r_2, r_3) = 15$ in T_1 and $\mathbb{C}(r_1, r_2) = 12$ in T_2, it yields $r_3 = 21$. Depending on the aggregation operator in use, it might not be so easy to infer the value of r_3 and maybe the attacker can only give an estimation or approximate value.

This same problem can arise in the case of dynamic data, so the operations to insert and remove elements should take it into account.

4 Dynamic microaggregation of VSM

We introduce our anonymization process by defining the insertion and deletion operations on the VSM. The data are statically anonymized (see[1]), and then records can be added or removed. The static anonymization through microaggregation can be described as a two step process:

- *Partition*: Define a partition P on the original data $\mathcal{D}$, where each cluster has at least k elements. This partition tries to ensure a minimum information loss.
- *Aggregation*: For each cluster $c_i \in P$, substitute each element in the cluster by its cluster representative or centroid. Usually, an aggregation operator $\mathbb{C}$ is used to compute the centroid $\hat{c_i} = \mathbb{C}(\{\boldsymbol{V(d_j)} \mid \boldsymbol{V(d_j)} \in c_i\})$.

We will use following cosine distance function between document vectors:

$$d(\boldsymbol{V}(d_1), \boldsymbol{V}(d_2)) = 1 - \frac{\boldsymbol{V}(d_1) \cdot \boldsymbol{V}(d_2)}{|\boldsymbol{V}(d_1)||\boldsymbol{V}(d_2)|} \tag{1}$$

where $\cdot$ is the dot product of the vectors. For the aggregation of the microaggregation step we use a component-wise mean to aggregate vectors:

$$\mathbb{C}(\{\boldsymbol{V}(d_1), \ldots \boldsymbol{V}(d_n)\}) = \frac{1}{n}(\sum_{i=1}^{n} w_{i,1}, \ldots, \sum_{i=1}^{n} w_{i,M}) \tag{2}$$

Depending on the concrete application other distances and aggregation operators could be used.

4.1 Simple document vector insertion

Given a k-anonymous VSM $\mathcal{D}' = \{\boldsymbol{V'}(d_1), \ldots, \boldsymbol{V'}(d_n)\}$, which forms a partition $P(\mathcal{D}') = \{c_1, \ldots, c_v\}$, we want to insert a new document vector $\boldsymbol{V}(d_*)$. $\hat{c_i}$ denotes the centroid of the cluster c_i. In order to do so we follow the procedure:

1. Find the cluster $c_i \in P(\mathcal{D}')$ such that $d(\boldsymbol{V}(d_*), \hat{c_i}) \leq d(\boldsymbol{V}(d_*), \hat{c_j})$ for all $c_j \in P(\mathcal{D}')$.
2. Add the new vector $\boldsymbol{V}(d_*)$ to the cluster c_i by applying the perturbation $\boldsymbol{V'}(d_*) = \hat{c_i}$ to the new vector.

The second step is important in order to prevent inference by intersection. Computing a new centroid for the cluster will reduce the information loss but will also lead to inference attacks as the one described in Section 3.1.

4.2 Simple document vector deletion

Given a k-anonymous VSM $\mathcal{D}' = \{\boldsymbol{V'}(d_1), \ldots, \boldsymbol{V'}(d_n)\}$, which forms a partition $P(\mathcal{D}') = \{c_1, \ldots, c_v\}$, we want to remove a document vector $\boldsymbol{V'}(d_*) \in \mathcal{D}'$. The deletion has the following steps:

1. Identify the cluster c_i such that $\boldsymbol{V'}(d_*) \in c_i$, and delete the vector from the cluster.
2. If $|c_i| < k$ find another cluster c_j such that $d(\hat{c_i}, \hat{c_j}) \leq d(\hat{c_i}, \hat{c_l})$ for all $c_l \in \mathcal{D}'$ and $l \neq i \neq j$.
3. Add the vectors of the cluster c_i to the cluster c_j. To do so all vectors $\boldsymbol{V'}(d_i) \in c_i$ are perturbed as $\boldsymbol{V'}(d_i) = \hat{c_j}$.

Steps 2 and 3 are equivalent to the insertion of elements, no new centroid is re-computed to avoid inference. There is however an important point to consider with respect to inference and deletion.

For example, Table 2 shows the deletion of the record r_6 from the 2-anonymous table T_1 resulting into table T_2. An attacker knowing both tables T_1 and T_2 will know that one of the records with value 30 has been deleted and the other has been merged into the cluster of records with value 25. No record can be identified or distinguished from the rest with probability higher than $1/k$. This is true if the user has no other knowledge from the records other than the quasi-identifiers.

Record	Age
r_1	10
r_2	10
r_3	21
r_4	21
r_5	30
r_6	30

(a) T_1

Record	Age
r_1	10
r_2	10
r_3	21
r_4	21
r_5	21

(b) T_2

Table 2: Example of deletion.

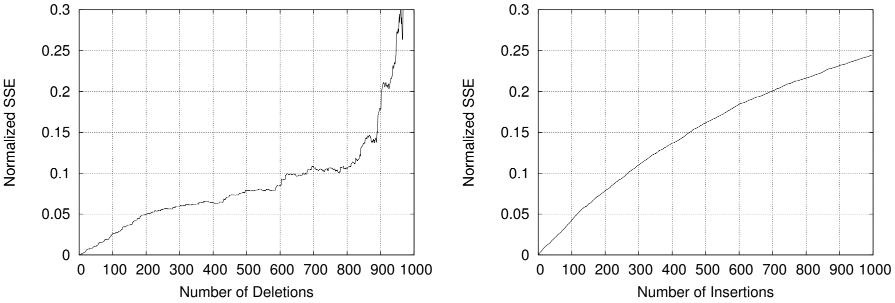
5 Evaluation

To provide an initial evaluation of the perturbation introduced by our proposal we rely mainly in observing the within cluster homogeneity (SSE). Given a protected VSM $\mathcal{D}' = \{\boldsymbol{V'}(d_1), \ldots, \boldsymbol{V'}(d_n)\}$, with a partition into clusters $P(\mathcal{D}') = \{c_1, \ldots, c_v\}$, and its respective original VSM $\mathcal{D} = \{\boldsymbol{V}(d_1), \ldots, \boldsymbol{V}(d_n)\}$, where $\boldsymbol{V'}(d_i)$ is the protected version of the vector $\boldsymbol{V}(d_i)$, we can compute an SSE as:

$$SSE(\mathcal{D}') = \sum_{c_i \in P(\mathcal{D}')} \sum_{\boldsymbol{V}(d_j) \in c_i} d^2(\boldsymbol{V}(d_j), \hat{c_i})$$

where d is the cosine distance. In order to compare sets of documents with different size, we will divide the SSE by the number of documents in the set to give a normalized SSE with respect to the number of documents.

We selected 1000 random documents from the R8 subset of the Reuters-21578 dataset [11], containing a collection of classified Reuters news. Stop-words are removed from the document as well as terms with two or less letters. Once the documents are cleaned we apply the Porter stemming algorithm [10], which considers all words with the same stem as the same word, producing a reduction in the size of the feature set.



(a) Deletions

(b) Insertions

Fig. 1: Normalized SSE for deletions and insertions on the 5-anonymous dataset.

From the initial set of documents we start by generating a 5-anonymous version of the set applying the MDAV algorithm [3]. With this set we first apply consecutive deletion of random elements and then, starting again from the protected 1000 set, we insert new documents. Figure 1 shows the evolution of the normalized SSE as elements are deleted or inserted from the 5-anonymous version of the dataset.

Note that the values are very low because SSE is divided by the number of elements in each case. The values of SSE for greatest number of insertions and deletions give an idea of the SSE value for maximum perturbation. So if we consider a maximum value of 0.3, then values of 0.1 and 0.05 represent approximately and respectively the 33% and 17%.

6 Conclusions

We have introduced the anonymization of a dynamic vector space model based on microaggregation. The vector space model can be used as the metadata of encrypted documents in a typical cloud storage service. The VSM is ensured to be k-anonymous with respect to the documents, and this property is maintained while documents can be inserted and deleted from the set. We have presented here an initial work which can be further developed. We plan to explore the application of other data privacy techniques and the improvement on reducing the information loss during insertions and deletions.

Acknowledgments

This work is partially funded by projects TSI2007-65406-C03-02, ARES-CONSOLIDER INGENIO 2010 CSD2007-00004, TIN2010-15764, and TIN2011-27076-C03-03 of the Spanish Government, and by project FP7/2007-2013 (Data without Boundaries). work contributed by one of the authors was carried out as part of the Computer Science Ph.D. program of the Universitat Autònoma de Barcelona (UAB).

References

1. D. Abril, G. Navarro-Arribas, V. Torra. Vector Space Model Anonymization. Sixteenth International Conference of the Catalan Association of Artificial Intelligence (CCIA 2013) (*to appear*).
2. J. W. Byun, Y. Sohn, E. Bertino, and N. Li, "Secure Anonymization for Incremental Datasets," in Secure Data Management, 2006, vol. 4165, pp. 48-63.
3. J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Trans. on Knowl. and Data Eng.*, 14:189–201, January 2002.
4. J. Cao, B. Carminati, E. Ferrari, and K.-L. Tan, "CASTLE: Continuously Anonymizing Data Streams," IEEE Transactions on Dependable and Secure Computing, vol. 8, no. 3, pp. 337-352, 2011.
5. S. De Capitani di Vimercati, S. Foresti, and G. Livraga, "Privacy in Data Publishing," in Data Privacy Management and Autonomous Spontaneous Security. LNCS 6514, Springer, 2011, pp. 8-21.
6. T. Iwuchukwu and J. F. Naughton, "K-anonymization as spatial indexing: toward scalable and incremental anonymization," in Proceedings of the 33rd international conference on Very large data bases, Vienna, Austria, 2007, pp. 746-757.
7. J. Li, B. C. Ooi, and W. Wang, "Anonymizing Streaming Data for Privacy Protection", in IEEE 24th International Conference on Data Engineering, 2008. ICDE 2008, 2008, pp. 1367-1369.
8. C.D. Manning, P. Raghavan, H. Schütze (2009) *An Introduction to Information Retrieval.* Cambridge University Press.
9. J. Pei, J. Xu, Z. Wang, W. Wang, and K. Wang, "Maintaining K-Anonymity against Incremental Updates," in 19th International Conference on Scientific and Statistical Database Management, 2007. SSBDM 07, 2007.
10. M.F. Porter. *An algorithm for suffix stripping.* Program Vol. 14, no. 3, pp 130–137, 1980.
11. Reuters Ltd., Reuters-21578, Distribution 1.0, 2004 http://www.daviddlewis.com/resources/testcollections/reuters21578
12. P. Samarati, "Protecting respondents identities in microdata release," IEEE Transactions on Knowledge and Data Engineering, vol. 13, no. 6, pp. 1010-1027, 2001.
13. K. Stokes and V. Torra, "Multiple releases of k-anonymous data sets and k-anonymous relational databases," International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 20, no. 06, pp. 839-853, Dec. 2012.
14. L. Sweeney, "k-anonymity: a model for protecting privacy," Int. J. Uncertain. Fuzziness Knowledge-Based Systems, vol. 10, no. 5, pp. 557-570, Oct. 2002.

15. T. M. Truta and A. Campan, ”K-anonymization incremental maintenance and optimization techniques,” in Proceedings of the 2007 ACM symposium on Applied computing, 2007, pp. 380-387.
16. X. Xiao and Y. Tao, ”M-invariance: towards privacy preserving re-publication of dynamic datasets,” in Proceedings of the 2007 ACM SIGMOD international conference on Management of data, 2007, pp. 689-700.
17. H. Zakerzadeh and S. L. Osborn, FAANST: Fast Anonymizing Algorithm for Numerical Streaming Data, in Data Privacy Management and Autonomous Spontaneous Security, LNCS 6514, 2011, pp. 36-50.