A Privacy Leakage Upper-bound Constraint based Approach for Cost-effective Privacy Preserving of Intermediate Datasets in Cloud (Supplementary File)

Xuyun Zhang, Chang Liu, Surya Nepal, Suraj Pandey, Jinjun Chen, *Member, IEEE*

APPENDIX A

A.1 Privacy Quantification for a Single Intermediate Dataset

In the real world, adversaries usually own certain background knowledge before they observe an intermediate dataset, which can be regarded as adversaries' prior belief [1], [2] and [3]. Without loss of generality, we just discuss the privacy quantification after observing an intermediate dataset without background knowledge.

We estimate $p(S = s, Q = q)$ (abbr. $p(s,q)$) via computing $p(S = s|Q = q)$ (abbr. $p(s|q)$), according to the relationship between them. The relationship is described as the following equation:

$$p(S = s|Q = q) = p(S = s, Q = q)/p(Q = q). \quad (1)$$

Then, we can compute the probability distribution $P^*(S,Q)$ from $P^*(S|Q)$ according to (1).

Similar to the setting in [1], we assume that the anonymized intermediate datasets satisfy l-diversity, i.e, all the quasi-identifiers are divided into a series of groups. Each group contains at least l different sensitive values in SD, where SD is the set of sensitive data. Let GI be the set of these groups, i.e., $GI = \{g_1, g_2, \dots, g_{|GI|}\}$. And let G be a random variable ranging within GI. Further, we assume that an intermediate dataset is part of original datasets as a result of either horizontal partition or vertical partition.

In terms of the analysis in [1] with maximum entropy principle [4], the posterior belief of $p(s,q)$ after observing intermediate dataset d^* can be directly deduced from the original dataset d, where $p(s,q)$ is the joint possibility of association $\langle s,q\rangle$. Applying the complete probability formula on $p(q,s)$, we can derive the following equation:

$$p(s,q) = p(s,q|\langle s,q\rangle \in d^*) \cdot p(\langle s,q\rangle \in d^*) + p(s,q|\langle s,q\rangle \in (d/d^*)) \cdot p(\langle s,q\rangle \in (d/d^*)). \quad (2)$$

Both $p(\langle s,q\rangle \in d^*)$ and $p(\langle s,q\rangle \in (d/d^*))$ can be calculated directly from the original dataset and the intermediate dataset, where d/d^* means the data items in d except that in d^*. Since no information of $\langle s,q\rangle$ that belongs to (d/d^*) can be inferred from dataset d^*, $p(s,q\,|\langle s,q\rangle \in (d/d^*))$ is assigned a value that makes the probability distribution of $\langle S,Q\rangle$ in (d/d^*) a uniform distribution. For conciseness, $p(s,q\,|\langle s,q\rangle \in d^*)$ is re-denoted as $p^*(s,q)$. The following complete probability formula on $p^*(s,q)$ is utilized to calculate it in terms of the observed dataset d^*:

$$p^*(s,q) = \sum_{i=1}^{|GI|} p^*(s,q \mid G = g_i) \cdot p(G = g_i). \quad (3)$$

The probability $p(G = g_i)$ can be computed by the proportion of quasi-identifiers in group g_i. We can estimate $p^*(s,q|G = g_i)$ via assigning a value that makes the probability distribution $\langle S,Q\rangle$ within group g_i a uniform distribution. Once the probability distribution $P^*(S,Q)$ of an original dataset is estimated from the intermediate dataset d^*, the privacy quantification of d^* can be defined as in the main file in Section 4.1 (Single Intermediate Dataset Privacy Representation).

APPENDIX B

B.1 Proof of Lemma 1

Lemma 1. *If $d_r \in D^{une}$, then for any $d \in PD(d_r)$, $d \in D^{une}$.*

Proof. For any $d_u \in D/(\{d_r\} \cup PD(d_r))$, $d_v \in PD(d_r)$, the inequality $PL_m(\{d_u, d_v\}) \le PL_m(\{d_u, d_r\})$ holds, i.e., the joint privacy leakage of an offspring dataset d_u and an dataset d_v is never larger than that of d_v and the root dataset d_r. In term of this interesting inequality, if the dataset d_r is unencrypted, i.e., $d_r \in D^{une}$, then all datasets of $SDT(d_r)$ can be left unencrypted without compromising privacy requirements. Hence, for any $d \in PD(d_r)$, $d \in D^{une}$. □

B.2 Proof of Lemma 2

Lemma 2. *Given a privacy-preserving solution $\langle D^{enc}, D^{une}\rangle$ with the minimal cost, a graph denoted as $EDG = \langle D^{enc} \cup \{d_o\}, E'\rangle$ must be a tree structure, where $E' \subseteq E$.*

Proof. We prove this lemma by contradiction. The graph *EDG* consists of all encrypted datasets and edges among them. Suppose that *EDG* is not a tree but a forest structure. Then, *EDG* contains at least two trees. Without loss of generality, we assume *EDG* contains two trees, T_1 and T_2, respectively. Since both T_1 and T_2 are subtrees of *SIT*, they must be connected by at least one dataset in D^{une}. Without loss of generality, we assume dataset d connects T_1 and T_2. Hence, either T_1 or T_2 must be part of $SDT(d)$. Suppose T_1 is part of $SDT(d)$, i.e., $T_1 \subseteq SDT(d)$. Since $d \in SDT(d)$, $SDT(d)$ should be $UST(d)$ according to the *RPC* property. As a result, the

datasets in T_1 should not be encrypted. This conclusion contradicts to the condition that $\langle D^{enc}, D^{une} \rangle$ is a privacy-preserving solution with the minimum cost. □

B.3 Proof of Theorem 1

Theorem 1. *Under the PLC$_1$ constraint, $\sum_{d \in UD_i} PL_s(d) \le \varepsilon_i$, $1 \le i \le H$, a construction process of a compressed tree on a SIT corresponds to a feasible global encryption solution π_f^k, i.e., the privacy leakage $PL_m(D)$ with regard to π_f^k can be ensured beneath the given threshold ε.*

Proof. As to the construction of compressed tree on a *SIT*, suppose the first i layers have been built up. Typically, it can be described by Fig.2. The compressed encrypted and unencrypted nodes are shown in Fig.2. The privacy leakage caused by the unencrypted datasets before layer L_{i+1} is determined by $\bigcup_{k=1}^{i} UD_k$. Because the construction process complies with the *PLC*$_1$ constraint, we can deduce the following inequality:

$$\begin{aligned} PL_m(\bigcup_{k=1}^{i} UD_k) &\le \sum_{k=1}^{i} PL_m(UD_k) \\ &\le \sum_{k=1}^{i} \sum_{d \in UD_k} PL_s(d) \\ &= \sum_{k=1}^{i} (\varepsilon_k - \varepsilon_{k+1}) \\ &= \varepsilon - \varepsilon_{i+1}. \end{aligned}$$

The privacy leakage of the datasets after L_i can be controlled under ε_{i+1} by the definition of ε_{i+1}. Then, the holistic privacy leakage of the *SIT* can never exceed the threshold ε according to the following inequality:

$$\begin{aligned} PL_m(D) &= PL_m\big(PD(ED_i) \cup \bigcup_{k=1}^{i} UD_k\big) \\ &\le PL_m\big(PD(ED_i)\big) + PL_m\big(\bigcup_{k=1}^{i} UD_k\big) \\ &\le \varepsilon_{i+1} + (\varepsilon - \varepsilon_{i+1}) \\ &= \varepsilon. \end{aligned}$$

Hence, during the whole construction of $CT(\pi_f^k)$, the privacy preserving can always be ensured. As a result, we can obtain a feasible global solution π_f^k on a *SIT* by constructing a compressed tree $CT(\pi_f^k)$, and the privacy of this solution can always be ensured. □

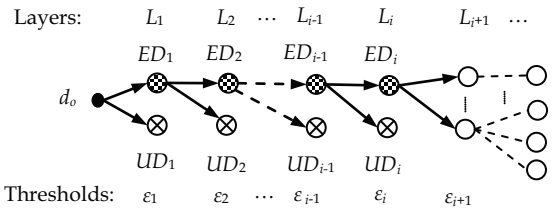

Fig.2. Construction of a compressed tree.

APPENDIX C

C.1 Recursive Minimum Privacy-Preserving Cost Algorithm

We exploit the Branch-and-Bound (B&B) technique [5] to desingn the optimal algorithm. The recursive algorithm, named as recursive minimum privacy-preserving cost algorithm (*R_MPPC*), is depicted in Algorithm 1. *SORT* and *SELECT* are two simple external functions. The optimal privacy-preserving cost can be obtained via invoking *R_MPPC*$(L_1, \varepsilon, 0)$. Before the algorithm is invoked, CDE_0 is initialized with $CD(d_o)$, i.e., $CDE_0 \leftarrow CD(d_o)$, and the variable minimum global cost $C_{minGlobal}$ is initially assigned with positive infinite: $C_{minGlobal} \leftarrow +\infty$.

ALGORITHM 2. **RECURSIVE MINIMUM PRIVACY-PRESERVING COST.**

Description	Recursively identifies the intermediate datasets need to be encrypted, achieving the minimal privacy-preserving cost under *PLC*$_1$.
Input	A *SIT* with root d_o; all attribute values of each intermediate dataset are given; current layer L_i, privacy leakage threshold ε_i; the current cost for a partial solution: $C_{current}$.
Output	A vector of local solutions $\langle \pi_i, \ldots, \pi_H \rangle$that comprise part of a potential optimal global solution; the minimum encryption cost after the layer L_{i-1}: $C_{minLocal}$.
Step 1	Check whether a feasible global solution is found via checking whether CDE_{i-1} is empty.
1.1 If $CDE_{i-1} = \emptyset$, it means a feasible global solution has been found. 1)The current cost $C_{current}$ now is the global cost of the feasible global solution. If $C_{current} < C_{minLocal}$, then $C_{minLocal}$ should be updated by $C_{current}$: $C_{current} \leftarrow C_{minLocal}$. Otherwise, go to Step 2. 2)Terminate this recursive invocation by returning cost 0. 1.2 Otherwise, go to Step 2.	
Step 2	Sort the elements in CDE_{i-1} in ascending order according to the hiding cost of each intermediate dataset: SORT(CDE_{i-1}).
Step 3	Generate all possible local solutions and search for the best one to constitute the optimal global privacy-preserving solution.
3.1 Generate all the possible local solutions: $\Lambda_i \leftarrow \langle x, CDE_{i-1}/x \rangle$, where $x \in 2^{CDE_{i-1}}$, $2^{CDE_{i-1}}$ is the power set of CDE_{i-1}. 3.2 Set the minimum local cost after the layer L_{i-1} with positive infinite: $C_{minLocal} \leftarrow +\infty$. 3.3 Select a solution from Λ_i: $\pi \leftarrow SELECT(\Lambda_i)$. A number with $\lvert CDE_{i-1} \rvert$ bits is employed to signify a local solution in the layer L_i. Each bit in this number corresponds to an element in the sorted CDE_{i-1}. For the value of each bit, 1 means the corresponding element is encrypted, and 0 means unencrypted. 3.4 Calculate the privacy leakage upper bound of π and the encryption cost: $PL_{local} \leftarrow \sum_{d \in UD_\pi} PL_s(d)$, $C_{local} \leftarrow \sum_{d_k \in ED_\pi} (S_k \cdot PR \cdot f_k)$. 3.5 Prune this branch if the privacy leakage of this solution exceed the threshold $PL_{local} > \varepsilon_i$, or the current cost is larger than the current minimum global cost $C_{local} + C_{current} > C_{minGlobal}$. 3.6 Calculate the remaining privacy leakage $\varepsilon_{i+1} \leftarrow \varepsilon_i - PL_{local}$, and then recursively invoke the algorithm to gain the minimum cost after the layer L_i: $C_{local} \leftarrow C_{local} + R\_MPPC\ (L_{i+1}, \varepsilon_{i+1}, C_{local} + C_{current})$. 3.7 Check whether the cost C_{local} is less than current $C_{minLocal}$. If yes, $C_{minLocal}$ should be updated using C_{local}: $C_{minLocal} \leftarrow C_{local}$. So do the current local optimal solution is set as π. 3.8 Delete π from Λ_i. If $\Lambda \neq \emptyset$, go to step 3.3 3.9 Return $C_{minLocal}$, the minimum cost of the layers after L_{i-1}.	

C.2 Time Complexity Analysis for the Optimal Algorithm

The optimal algorithm suffers from poor efficiency when there are a large number of intermediate datasets due to its huge state-search space. The search tree generated from a *SIT* is of huge size, as its node that corresponds to a local privacy-preserving solution $\pi_i = \langle ED_i, UD_i \rangle$ has $2^{|CDE_{i+1}|}$ child nodes to be checked. Therefore, the time consumption will increase exponentially when the num-

ber of intermediate datasets is getting larger. Mathematically, we analyze the time complexity of this optimal algorithm as follows.

Suppose that a *SIT* has n intermediate datasets, i.e., $|D| = n$, and the number of intermediate datasets in the layer L_i is denoted as n_i. Thus, $\sum_{i=1}^{H} n_i = n$, where H is the height of the *SIT*. Let m_i be the number of intermediate datasets in CDE_{i-1}. Then, $m_i \leq n_i$, and the equation holds if all the intermediate datasets in L_{i-1} are hidden. Let $T(m_i)$ be the running time of this algorithm in layer L_i. Then, we have $T(m_i) = 2^{m_i}$ as analyzed above. Thus, we can derive that $T(m_i) = O(2^{m_i})$. Let $T(n)$ be the total running time. As a consequence of recursive invocation, $T(n) = \prod_{i=1}^{H} T(m_i)$. Further, we have $T(n) = O(2^{\sum_{i=1}^{H} m_i})$. In the worst case, $m_i = n_i$. Thus, we can conclude that $T(n) = O(2^{\sum_{i=1}^{H} m_i}) = O(2^n)$. Although the running time of the optimal algorithm is exponential in the worst case, the average performance can be better due to the adoption of Branch-and-Bound pruning techniques and sorting the elements in advance. However, this algorithm is much better than the exhaustive approach in which all the subsets of D will be enumerated and checked, and the corresponding time complexity will be $\Theta(2^n)$.

APPENDIX D

D.1 Details about Real-world Datasets

We use the Adult dataset from UCI Machine Learning Repository [6], which is a popular common-used dataset in privacy research community. After pre-processing the Adult dataset, the original dataset consists of 18,909 records, i.e., $|QI|$ = 18,909. Each record has 14 attributes. The attribute *Education* is utilized as the sensitive attribute. This sensitive attribute has 16 different categorical values, i.e., $|SD|$ = 16.

Twelve datasets are generated from the original dataset. These intermediate datasets constitute a ternary tree with two levels. The children datasets are directly generated from their parent datasets. Then we implement the generalizing algorithm proposed in [7] to anonymize the intermediate datasets to satisfy 4-diversity. The privacy leakage of each intermediate dataset is quantified as formulated in Section 4.1. The values of the dataset size and privacy leakage are summarized in Table 1. Note that the values of privacy leakage are roughly proportional to dataset size because all the attributes are taken into account. Yet, in real-world scenarios, privacy leakage is less relevant to dataset size because the values of attributes that are not included in quasi-identifier can vary quite a lot. The frequency of each intermediate dataset is generated randomly in the interval [10, 100] according to the uniform distribution.

TABLE 1 DATASET SUMMARY FOR THE REAL-WORLD DATASETS.

Dataset	d_1	d_2	d_3	d_4	d_5	d_6
Size (M)	1.707	1.134	1.514	1.277	0.863	1.145
$PL_8(\cdot)$	1.809	1.201	1.604	1.353	0.914	1.213
Dataset	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}
Size (M)	0.848	0.570	0.742	1.136	0.762	1.007
$PL_8(\cdot)$	0.898	0.604	0.785	1.204	0.807	1.066

REFERENCES

[1] W. Du, Z. Teng and Z. Zhu, "Privacy-Maxent: Integrating Background Knowledge in Privacy Quantification," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD'08)*, pp. 459-472, 2008.

[2] T. Li, N. Li and J. Zhang, "Modeling and Integrating Background Knowledge in Data Anonymization," *Proc. 25th IEEE Int'l Conf. Data Engineering (ICDE'09)*, pp. 6-17, 2009.

[3] D.J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke and J.Y. Halpern, "Worst-Case Background Knowledge for Privacy-Preserving Data Publishing," *Proc. 23rd IEEE Int'l Conf. Data Engineering (ICDE'07)*, pp. 126-135, 2007.

[4] E.T. Jaynes, "Information Theory and Statistical Mechanics," *Physical Review*, vol. 106, no. 4, pp. 620-630, 1957.

[5] L.G. Mitten, "Branch-and-Bound Methods: General Formulation and Properties," *Operations Research*, vol. 18, no. 1, pp. 24-34, 1970.

[6] UCI Machine Learning Repository, ftp://ftp.ics.uci.edu/pub/machine-learning-databases/, accessed on: June 01, 2012.

[7] B.C.M. Fung, K. Wang and P.S. Yu, "Anonymizing Classification Data for Privacy Preservation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 5, pp. 711-725, 2007.