Privacy-Preserving Multi-Keyword Search in Information Networks

Yuzhe Tang, Ling Liu, *Fellow, IEEE*

Abstract—In emerging multi-domain cloud computing, it is crucially important to provide efficient search on distributed documents while preserving their owners' privacy, for which privacy preserving indexes or PPI presents a possible solution. An understudied problem for PPI techniques is how to provide *differentiated* privacy preservation in the face of multi-keyword document search. The differentiation is necessary as terms and phrases bear innate differences in their meanings.
In this paper we present ϵ-MPPI, the first work on distributed document search with quantitative privacy preservation. In the design of ϵ-MPPI, we identified a suite of challenging problems and proposed novel solutions. For one, we formulated the quantitative privacy computation as an optimization problem that strikes a balance between privacy preservation and search efficiency. We also addressed the challenging problem of secure ϵ-MPPI construction in the multi-domain network which lacks mutual trusts between the domains. Towards a secure ϵ-MPPI construction with practical performance, we proposed techniques for improved performance of secure computations by making a novel use of secret sharing. We implemented the ϵ-MPPI construction protocol with a functioning prototype. We conducted extensive experiments to evaluate the prototype's effectiveness and efficiency based on a real-world dataset.

Index Terms—Privacy, secure computation, multi-domain clouds, data indexing, distributed systems.

1 Introduction

In the age of cloud computing, data users, while enjoying a multitude of benefits from the cloud (e.g. cost effectiveness and data availability), are simultaneously reluctant or even resilient to use the clouds, as they lose data control. The recent research and industrial efforts towards returning data control back to cloud users have given birth to a variety of multi-domain cloud platforms, most notably emerging *information networks*. In an information network, a data owner can retain the full control of her data by being able to choose from an array of service providers one that she can presumably trust or even be able to launch a personal server administrated directly by herself. The information network does not need mutual trusts between servers, that is, an owner only needs to trust her personal server and nothing more.

Information networks emerge in a variety of application areas. For an example, in the enterprise intranet (e.g. IBM YouServ system [1], [2]), employees can store and manage their own documents on personally administrated machines. While the employees have their personal privacy concerns and could set up access control policies on the local documents, they may be required by the corporate-level management team to share certain information for the sake of promoting potential collaborations [2]. For another example, several distributed social networks (e.g. Diaspora [3], Status [4] and Persona [5]) recently emerge and become increasingly popular, which are based on the design of decoupling the storage of social information and social networking functionality. Unlike the centralized monolithic social networking (e.g. Facebook and LinkedIn), the distributed social networks allow an average social user to launch a personal server for storing her own social data and enforcing self-defined access control rules for privacy-aware information sharing [6]. Other examples of information networks include electronic Healthcare over the public Internet (e.g. the open-source NHIN Direct project [7]), peer-to-peer file sharing with access controls [8] and others.

In all these networks, a data owner can have an exclusive domain for administration of physical resources (e.g., a virtual machine) and data management of personal data under the full user control. Domains located inside multiple servers are isolated and distrusted between each other.[1] Information sharing and exchanges across a domain boundary are desirable for various application needs.

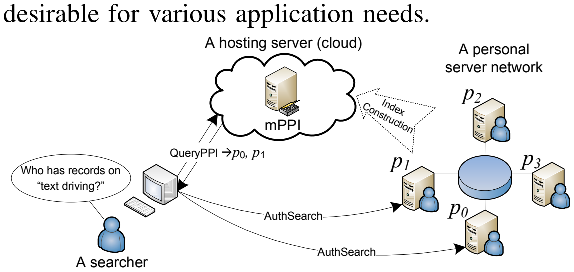


Fig. 1: The PPI system

- *Y.Tang is with the EECS department at Syracuse University, 4-206 Center for Science and Technology, Syracuse, NY 13244-4100 USA. E-mail: ytang100@syr.edu*
- *L. Liu is with the College of Computing, Georgia Institute of Technology, 266 Ferst Dr, Atlanta, GA 30332-0765 USA. E-mail: ling.liu@cc.gatech.edu*

1. In this paper, an owner is associated to one distinct domain and owns one (virtual) server in the network. We use the notation of an owner, a domain and a server, interchangeably.

For privacy-aware search and information sharing in the information networks, a candidate solution is a privacy preserving index on access controlled distributed documents [9], [10], [11], or PPI for short. In Figure 1, a PPI is a directory service hosted in a third-party entity (e.g. a public Cloud) that serves the global data to a number of data consumers or searchers. To find documents of interest, a searcher would engage in a two-stage search procedure: First she poses a query of relevant keywords against the PPI server, which returns a list of candidate owners (e.g. p_0 and p_1) in the network. Then for each candidate owner in the list, the searcher contacts its server and requests for user authentication and authorization before searching locally there. Note that the authentication and authorization only occur inside the information network, but not on the PPI server.

Comparing to existing work on secure data serving in the cloud [12], [13], [14], the PPI scheme is unique in the sense that 1) Data is stored in plain-text (i.e. without encryption) in the PPI server, which makes it possible for efficient and scalable data serving with rich functionality. Without use of encryption, PPI preserves user privacy by adding noises to obscure the sensitive ground truth information. 2) Only coarse-grained information (e.g. the possession of a searched phrase by an owner) is stored in the PPI server, while the original content which is private is still maintained and protected in the personal servers, under the user-specified access control rules.

Differentiating the Privacy Preservation of Multi-term Phrases

In the PPI system, it is desirable to provide differentiated privacy preservation regarding different search phrases and owners. The data model (elaborated in § 2) used in a PPI system and an information network is that each server possesses multiple documents, each consisting multiple terms. What is deemed private and should be protected by a PPI is the possession information in the form of "whether an owner possesses at least one document relevant to a multi-term phrase [2] ". Under this model, the meaning of differentiated privacy preservation is of two folds: 1) Different (single) terms are not born equal in terms of how sensitive they are. For example, in an eHealthcare network, it is natural for a woman to consider her medical record of an "abortion" operation to be much more sensitive than that of a "cough" treatment. 2) A multi-term phrase, as a semantic unit, can be much more (or less) sensitive than a single term contained in the phrase. For instance, "text" and "driving" are two terms that may be deemed non-sensitive in their solitary appearances, but a record of "text driving" can be considered more sensitive.

The existing PPI work [9], [10], [11], while designed to protect privacy, is not able to differentiate privacy preservation on different terms. Due to the quality-agnostic methods used for constructing these PPIs, they can not deliver a quantitative guarantee for privacy preservation for search of a single term, let alone that of a multi-keyword phrase.

In this paper, we propose ϵ-MPPI, a new PPI abstraction which can quantitatively control the privacy leakage for multi-keyword document search. In the ϵ-MPPI system, different phrases, be it either a single term or a multi-term phrase, can be configured with an intended degree on privacy, denoted by ϵ. ϵ can be of any value from 0 to 1; Value 0 represents the least concern on privacy preservation, while value 1 aims at the best privacy preservation (potentially at the expense of extra search overheads). By this means, an attacker, searching a multi-term phrase on ϵ-MPPI, can only have the confidence of mounting successful attacks bounded by what the phrase's privacy degree allows.

Constructing an ϵ-MPPI from an information network is challenging from the angles of both the computation and system designs. Computationally, the ϵ-MPPI construction requires careful design to properly add false positives (i.e. an owner who does not possess a term or a phrase falsely claims to possess it) so that a true positive owner can be hidden among the false positive ones, thus preserving privacy.

In terms of system designs, in a real information network which lacks mutual trusts between autonomously operated servers, it is necessary and desirable to construct ϵ-MPPI *securely* without a trusted authority. The task of distributed secure ϵ-MPPI construction would be very challenging. On one hand, constructing ϵ-MPPI to meet the stringent privacy constraints under a number of multi-term searches while minimizing extra search costs can be essentially modeled as an optimization problem, solving which requires complex computations such as a non-linear programming or NLP. On the other hand, while the common wisdom for secure computations (as required by the secure ϵ-MPPI construction) is to use a multi-party computation technique or MPC [15], [16], [17], [18] which protects input data privacy, the existing MPC techniques can work pragmatically well only with a simple workload in a small network. For example, FairplayMP [16], a representative practical MPC platform, "needs about 10 seconds to evaluate (very simple) functions" [19] which can otherwise be done within milliseconds by the regular non-secure computation. Directly applying the MPC techniques to the ϵ-MPPI construction problem which involves a complex computation and a large number of personal servers could lead to a cost that is truly spectacular and practically unacceptable.

To address the challenges of efficient secure ϵ-MPPI construction, our core idea is to draw a line between the secure part and non-secure part in the computation model. We minimize the secure computation part as much as possible by exploring various techniques (e.g. computation reordering). By this way, we have successfully separated the complex NLP computation from the MPC part such that the expensive MPC in our ϵ-MPPI construction protocol only applies to a very simple computational task, thus optimizing overall system performance.

The contributions of this paper can be summarized as following.

2. In this paper, we use "term" and "keyword" interchangeably.

- We proposed ϵ-MPPI to address the needs of differentiated privacy protection of multi-term phrases in a PPI system. To best of our knowledge, ϵ-MPPI is the first work on the problem. ϵ-MPPI guarantees the quantitative privacy protection by carefully controlling the false positives in a PPI and thus effectively limiting an attacker's confidence.
- We proposed a suite of practical ϵ-MPPI construction protocols applicable to the network of mutually untrusted personal servers. We specifically considered both the single-term and multi-term phrase cases, and optimized the performance of the secure ϵ-MPPI construction from both angles of computation model and system design by exploring the ideas of simplifying the secure computation tasks as much as possible while without sacrificing the quality of privacy preservation.
- We implemented a functioning prototype for ϵ-MPPI, based on which an experiment study confirms the performance advantage of our index construction protocol.

2 The ϵ-MPPI Model

2.1 Data and System Model

In an information network, each individual owner virtually owns a private domain p_i in which physical resources (e.g. a machine or a virtual machine) are fully administrated by the owner or by someone the owner presumably trusts. In our model, we consider m such owners in the network. In a domain, the owner maintains unstructured data, mostly a collection of documents consisting of multiple terms. We denote a term by t_j, and there are totally n terms in the vocabulary. For example, for media file sharing applications, such documents can be text describing the original media files, or in other applications, it could be personal medical records or social presence data. For an owner, all her personal documents are protected under access control rules defined by herself. Since the domain is fully managed by the owner, it is trivial to enforce the access rules. For search efficiency, an inverted index may be constructed locally inside each owner's domain. We abstract the content of an owner by a list of terms contained in the documents of the owner. This list, called *local vector*, has each element to describe the possession of a term by this owner. For example, List $\langle t_1 : 0, t_0 : 1\rangle$ owned by p_i indicates that p_i possesses some documents containing term t_0, but it is not the case for term t_1.

On the searcher side, our query model is a series of queries, each as a multi-term phrase. We denote a multi-term phrase by r_k where k is the phrase index, and we consider l phrases/queries in total, that is, $k \in [0, l-1]$. A query on phrase r_k needs to return all documents distributed in the network that are relevant to r_k. In practice, a queried phrase consists of fewer than 7 terms. Associated with each query r_k, we assume an intended privacy protection degree, denoted by ϵ_k, upon which everyone in the ϵ-MPPI system agrees.[3]

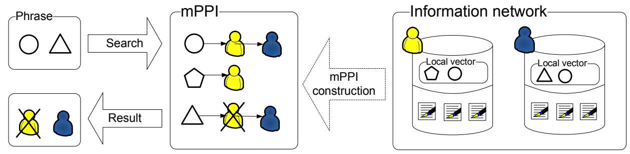


Fig. 2: The PPI data model: Different shapes in the local vector represent different indexed terms and ϵ-MPPI maintains the mapping from terms to data owners. $\times$ represents a false positive owner in ϵ-MPPI, which actually does not possess term $\triangle$.

To carry out the query, we consider a two-stage search-and-then-authorize process. Query on phrase r_k is first sent to the third-party ϵ-MPPI server, which will then redirect the query to all servers whose local vectors match r_k, that is, the corresponding element is 1. After that, each forwarded server authorizes the searcher and then uses local inverted index to find relevant documents. We stress that our query is for document/resource discovery in an untrusted environment. The document discovery is different from the traditional search which occurs between two trusted entities and has to assume trust relationship established in advance; for instance in social networks, a social user's search is forwarded to her trusted friends. In our case, there is no trust between the searcher and searched servers, which allows searchers to freely discover more potential documents of interest owned by people yet to trust.

To perform the forwarding in the search process, we employ an ϵ-MPPI which is internally structured as a coarse-grained inverted index, that is, the indexing occurs at the owner/server level rather than the document level. Figure 2 shows the intuition. Conceptually, the index can be modeled by an term-to-owner incident matrix, denoted by $M(i, j)$, in which a row and column represent an owner and an indexed term respectively, and a cell, say at row i and column j, is a binary value 0 or 1, which indicates whether owner p_i possesses content relevant to term t_j. The published ϵ-MPPI data, denoted by $M'(i, j)$, is similar to the ground-truth data, $M(i, j)$, except for the added noises. For an ϵ-MPPI server, the possession data is useful for redirecting search requests; given queried phrase r_k, ϵ-MPPI redirects to all candidate owners p_i's such that $M'(i, j) = 1$ for $\forall j \ a_{k,j} = 1$. The notations are summarized in Table 1.

2.2 Attack Model

Our threat model considers an attacker who can make a probabilistic claim about the sensitive possession fact that "owner p_i possesses a phrase r_k". For phrase r_k, this fact amounts to that $M(i, j) = 1$, for $\forall j \ a_{k,j} = 1$. Knowing such fact may disclose data owner privacy especially when the relevant data is inaccessible to the searcher and not supposed to be known by the searcher. Recall the previous

3. Choosing a proper sensitivity degree for a phrase (namely ϵ_k) is dependent on specific applications, and is a problem out of this paper's scope, while it is addressed by other recent work [20].

example; a person may not want to disclose her records of "text driving", since disclosing them could harm her chance in the job market or even causes a legal issue.

To choose a vulnerable owner and phrase to attack, the attacker can employ different strategies and exploit various information. Here, we consider an information-flow model in which exploitable information flows from data sources to the attacker through channels; the data sources and channels depend on the role and capability of the attacker. We consider the following three situations:

A **Knowing the public PPI (M')**: The attacker can learn the published PPI data (i.e. M') through a public channel, because $M'(i,j)$ is in a public domain and made available to anyone without authentication. Through a regular PPI search on the PPI server, one can exploit $M'(i,j)$ and makes a claim on the ground truth of phrase r_k possessed by owner p_i (if $M'(i,j) = 1$). The attacker may launch a series of searches on multiple phrases and combine the search results to improve the attack success ratio on a single phrase. For example, the attacker can choose to first search a multi-term phrase , say $t_1 t_0$, and then search a single-term included in it, say t_1.

B **Knowing the ground truth (M)**: The attacker can know certain part of ground truth data M. It can be disclosed through two possible situations: B.1) accessible part of data on a server (in this case the attacker is a regular searcher), and B.2) collusion with the owner so that the owner's ground-truth data is known to the searcher. The information can further help the attacker learn other part of M about inaccessible owners.

C **Knowing the index construction**: The attacker can learn information revealed during the index construction process. This is possible because the attacker can collude with owners and in the index construction process owners have to exchange information and thus may potentially learn others' private information.

2.3 Privacy Metric and Degree

We measure the privacy disclosure by the attacker's confidence in the success of an attack. Given that the PPI search results are mixed with false positives, the attacker's confidence translates into the probability that the attacker can successfully pinpoint a true positive provider from a search result. For an extreme example, if the search results are all true positives (and the attacker knows that), the privacy should be considered to be disclosed with a high degree. Formally, we define the privacy degree to be $Pr(M|M', O)$; here M is inaccessible content to the searcher, and O is exploitable information sources as previously described.

TABLE 1: Notations

p_i	The i-th owner	m	Number of owners
t_j	The j-th term	n	Number of terms
r_k	The k-th phrase	l	Number of phrases
ϵ_k	Privacy degree for phrase t_k	ϵ'_j	Calculated privacy degree for term t_j
σ'_k	Frequency of phrase r_k	σ_j	Frequency of term t_j
M	Term-incidence matrix	M'	Published term-incidence matrix
$A[a_{k,j}]$	Membership of a term in a phrase (be it present or non-present)		

Based on our information flow model, we define a series of privacy degrees which capture different levels of privacy leakage:

- **UNLEAKED**: The information can not be leaked from the source, thus making any attacks unsuccessful. This is the highest level of privacy preservation, since we essentially prohibit any information flows to the attacker, leaving her possible attacks baseless.
- **NOGUARANTEE**: The information can flow to the attacker but there is no guarantee on achieved privacy degree, rendering the privacy leakage level unpredictable.
- **FIXEDPROTECTION**: In this case, the system may guarantee privacy degree, but the privacy guarantee is at fixed values, and can not be configurable if users prefer a privacy degree that is not provided.
- **ϵ-PHRASE-PRIVACY**: Users can control privacy protection to achieve the desired level. The privacy protection is measured by the metrics described as above, and the PPI system provides guarantees that configured value of privacy degree must be achieved. Formally, a PPI is with degree ϵ-PHRASE-PRIVACY, if and only if for any phrase r_k with pre-configured privacy degree ϵ_k, the following inequality holds.

$$Pr(M_k|M'_k, O) \leq 1 - \epsilon_k \quad (1)$$

In this work, we mainly consider a statistical guarantee, that is, the actual privacy protection is *expected* to be better than the user-configured protection level. In our technical report [21], we propose extensions with finer-grained control of the quality of privacy preservation.

Note that we consider a multi-term search as a single unit for user-privacy configuration; that is, multiple phrases, while maybe overlapping, are configured separately in its privacy preservation. Our privacy preserving mechanism is designed to guarantee that the achieved false positive rate will meet privacy configurations of all relevant phrases.

2.4 ϵ-MPPI Privacy Guarantees

In our ϵ-MPPI design, we consider all three attack types described above. We consider that an attacker can mount one particular type or multiple types of attacks in order to gain higher success ratio. We show the privacy degrees achievable in our ϵ-MPPI under different attack circumstances, as shown in Table 2. We will analyze those privacy guarantees in details (§ 6). For the record, this paper focuses on the privacy and information confidentiality, and does not particularly address authenticity or other security properties.

TABLE 2: Attacks and achieved privacy degree

Attacks				Privacy degrees in ϵ-MPPI
A	B.1	B.2	C	
+				ϵ-PHRASE-PRIVACY
	+			ϵ-PHRASE-PRIVACY
			+	ϵ-PHRASE-PRIVACY
+	+			ϵ-PHRASE-PRIVACY
+	+		+	FIXEDPROTECTION
+		+		FIXEDPROTECTION
+		+	+	FIXEDPROTECTION

3 ϵ-MPPI CONSTRUCTION OVERVIEW

In this section, we present an overview for the computation model of ϵ-MPPI construction.

The ϵ-MPPI construction can be modeled as a process consisting two stages: a multi-source analytical computation and a randomized publication. Given privacy degree $\{\epsilon_k\}$ and ground-truth information M, the multi-source analytical computation produces a number of probability values, denoted by $\{\beta\}$. Then the randomized publication process leverages the probabilities to randomly add false positives for publishing each owner's local vector. To be specific, given a β value for term t_j or phrase r_k, the randomized process is essentially to flip the binary elements in the local vector based on the following formula.

$$
\begin{aligned}
0 &\rightarrow \begin{cases} 1, \text{ with probability } \beta \\ 0, \text{ otherwise} \end{cases} \\
1 &\rightarrow 1
\end{aligned} \tag{2}
$$

In this formula, the input value is from ground-truth possession data M and the output is the published data in M'. When the input is 1 meaning that an owner does possess the term or phrase, it is always published to be 1 in M'. Here, this truthful publication rule can guarantee a true positive owner is always included in a relevant search result, thus ensuring a 100% search recall. When the input is 0 meaning an owner does not possess the term, it is flipped to be 1 with probability β. This untruthful publication rule adds false positive owners in the published PPI for obscuring the identities of true positive ones. Note that the false positives, while preserving privacy, may cause additional search cost and decrease search precision. We illustrate the PPI publication framework in Figure 3, where two false positives (i.e. the underlined number) are added in the published PPI. For term t_0, with $\beta = 0.5$, one out of two negative owners (i.e. the two 0s in the first row in the figure) are expected to be flipped to 1 as a false positive.

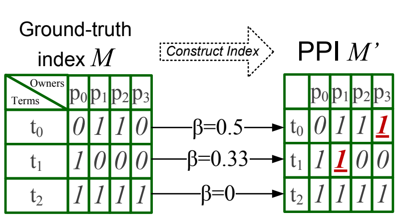


Fig. 3: PPI publication with probability β's

Single-term publication: Under this framework, the key is the first stage, that is, to carry out the multi-source analytics and compute β properly for the quality of privacy preservation. We start with publishing the single-terms individually. In this case, each β should be associated with one term. It needs to compute β large enough to make the expected number of flipped binary values (or false positives) be bigger than the desired level of privacy preservation, that is, $\epsilon_j \cdot m(1-\sigma_j)$. We have the following equation:

$$
\begin{aligned}
\epsilon_j &= \frac{(1-\sigma_j)\cdot\beta_b(t_j)}{(1-\sigma_j)\cdot\beta_b(t_j)+\sigma_j} \\
\Rightarrow \beta_b(t_j) &= [(\sigma_j^{-1}-1)(\epsilon_j^{-1}-1)]^{-1}
\end{aligned} \tag{3}
$$

More challenging is to handle the multi-term phrases. We propose two paradigms for β computations for multiple terms (§ 4): 1) a single term-oriented publication in which we re-use the single-term publication process for the phrase publication. Specifically, *per-term* β's are produced by the multi-source analytical process and different terms get published independently in the randomized process, 2) a phrase-wise publication in which *per-phrase* β's are produced by the multi-source analytical process and different terms are published in a correlated fashion. We further propose two approaches under the single term-oriented publication, called `MaxE`(§ 4.1.1) and `ENLP`(§ 4.1.3) respectively. We propose one approach for the phrase-wise publication, called `IBeta`(§ 4.2.1). Our ϵ-MPPI design is also capable of handling new server joins, as elaborated in our technical report [21].

We realize the computation framework securely over the information network without mutual trusts (§ 5).

4 PUBLISHING ϵ-MPPI

Our ϵ-MPPI is designed to be configurable on a per-phrase basis. Multi-term publication in ϵ-MPPI is built upon the single-term publication. We proposed two general approaches for such extensions: single term-oriented publication and phrase-wise publication.

4.1 Single Term-Oriented Publication

For a single term-oriented publication, our idea is to reuse the single term publication process in a multi-term context. To be specific, we treat the single-term publication process as a black box, and convert per-phrase privacy degree ϵ_k to the per-term degree ϵ'_j as the input to single-term publication process. We propose two specific approaches to convert the per-phrase ϵ to the per-term ϵ'.

4.1.1 `MaxE`: A Basic Heuristic

A basic heuristic to generate the per-term degree (ϵ'), which we call `MaxE`, is to *augment* a term's privacy degree to be bigger than the privacy degrees of any private phrases that cover the term. To be specific, given term t_j and a set of phrases that contains t_j, say $\{r_k\}$, the per-term privacy degree, ϵ'_j, is set to be the maximal among all the private degrees, $\{\epsilon_k\}$, as below:

$$\epsilon'_j = \max_{\forall kk \text{ s.t. } a_{j,kk}=1} (\epsilon_{kk}) \tag{4}$$

For example, given terms t_0 and t_1, if the per-phrase privacy degrees are $\epsilon_0 = 0.4$ for two-term phrase $r_0 = t_1 t_0$, $\epsilon_1 = 0.3$ for single-term phrase t_0 and $\epsilon_2 = 0.5$ for single-term phrase t_1, then using Equation 4 we have $\epsilon'_0 = \max(\epsilon_1, \epsilon_0) = 0.4$ and $\epsilon'_1 = \max(\epsilon_2, \epsilon_0) = 0.5$.

The intuition behind `MaxE`is that when publishing on a per-term basis, it can guarantee that *both* per-term privacy and relevant per-phrase privacy are protected. We analyze how well the `MaxE` approach preserves the multi-term privacy. First, since the `MaxE` approach publishes multiple terms independently, we have the following theorem.

Theorem 4.1: When terms are independently distributed on owners, the single term-oriented publication can guarantee (statistically) that the false positive rate, that is, $1 - Pr(M_k|M'_k)$ for multi-term phrase r_k, is as big as the false positive rate of any single term in the phrase. That is,

$$1 - Pr(M_k|M'_k) \geq \max_{\forall j \text{ s.t. } a_{k,j}=1} (1 - Pr(M_j|M'_j)) \tag{5}$$

Proof: The proof is based on mathematical induction. We start with the base case where two terms, t_0 and t_1, are considered. Since terms are independent, the frequency of the two-term phrase, "$t_0 t_1$", is the multiplication of those of each term, that is, $\sigma_{(0,1)} = \sigma_0 \sigma_1$. Terms are independently published and the total false positives of phrase "$t_0 t_1$" come from three sources, and the number, denoted by $F_{(0,1)}$, is expected to be $F_{(0,1)} = \sigma_0(1-\sigma_1)\times\beta_1 + (1-\sigma_0)\sigma_1\times\beta_0 + (1-\sigma_0)(1-\sigma_1)\times\beta_0\beta_1$, leading to the multi-term false positive as below. Here we use $fp_{(\cdot)}$ to denote the false positive rate, such as, $fp_k = 1 - Pr(M_k|M'_k)$.

$$\begin{aligned} fp_{(0,1)} &= \frac{F_{(0,1)}}{F_{(0,1)} + \sigma_0\sigma_1} \\ &= 1 - \frac{1}{1 + (\sigma_0^{-1} - 1)\beta_0} \cdot \frac{1}{1 + (\sigma_1^{-1} - 1)\beta_1} \end{aligned} \tag{6}$$

In the single-term case, say term t_0, given publishing probability β_0, false positive rate fp_0 is expected to be,

$$fp_0 = 1 - \frac{1}{1 + (\sigma_0^{-1} - 1)\beta_0} \tag{7}$$

Similarly,

$$fp_1 = 1 - \frac{1}{1 + (\sigma_1^{-1} - 1)\beta_1} \tag{8}$$

Plugging Equation 7, 8 in Equation 6, it yields,

$$\begin{aligned} fp_{(0,1)} &= 1 - (1 - fp_0)(1 - fp_1) \\ &= fp_1 + fp_0(1 - fp_1) \end{aligned} \tag{9}$$

Since $fp_0, fp_1 \in [0,1]$, $fp_0(1 - fp_1) \geq 0$ and $fp_{(0,1)} \geq fp_1$. By symmetry, $fp_{(0,1)} \geq fp_0$, and thus, we have,

$$fp_{(0,1)} \geq \max(fp_0, fp_1)$$

Now suppose it holds that $fp_{(0,\ldots,k-1)} \geq \max_{j=0,\ldots,k-1} fp_j$. Consider the case for $fp_{(0,\ldots,k)}$. Publishing based on term t_k and phrase "$t_0 \ldots t_{k-1}$" is similar to that based on t_1 and t_0. Thus by applying Equation 9, we have $fp_{(0,\ldots,k)} = (1 - fp_k)fp_{(0,\ldots,k-1)} + fp_k$. It yields that $fp_{(0,\ldots,k)} > \max(fp_{(0,\ldots,k-1)}, fp_k)$. Based on the inductive hypothesis,we can get,

$$fp_{(0,\ldots,k)} \geq \max_{j=0,\ldots,k} fp_j$$

$\square$

Since the single-term publication guarantees that $1 - Pr(M_j|M'_j) \geq \epsilon'_j$. Thus by plugging Equation 4 in Equation 5, we have,

$$\begin{aligned} 1 - Pr(M_k|M'_k) &\geq \max_{\forall j \text{ s.t. } a_{k,j}=1} (1 - Pr(M_j|M'_j)) \\ &\geq \max_{\forall j \text{ s.t. } a_{k,j}=1} (\epsilon'_j) \\ &\geq \max_{\forall j \text{ s.t. } a_{k,j}=1} \Big(\max_{\forall kk \text{ s.t. } a_{j,kk}=1} (\epsilon_{kk}) \Big) \\ &\geq \max_{\forall j \text{ s.t. } a_{k,j}=1} \Big(\epsilon_k \Big) = \epsilon_k \end{aligned} \tag{10}$$

Note that the last step is due to that $a_{k,j} = 1$. Equation 10 means that multi-term privacy degree can be guaranteed as long as the single term privacy can be guaranteed and terms are distributed independently.

4.1.2 Extending `MaxE` for Correlated Terms

In order to handle the case where terms are correlated in distribution, we extend the `MaxE` protocol with additional requirements. Here we mainly consider two terms, say t_0 and t_1; for phrases with more than two terms they can be recursively broken down to multiple two-term phrases. Given frequencies of two terms [4], δ_0 and δ_1, we require that the publication probability β_0 and β_1 should follow:

$$\frac{\beta_0}{\beta_1} = \frac{\delta_0}{\delta_1} \tag{11}$$

$$\begin{aligned} \beta_0 &\geq \frac{1}{\delta_1}\Big(\delta_{(0,1)} - \frac{(\delta_0 - \delta_{(0,1)})(\delta_1 - \delta_{(0,1)})}{1 - \delta_0 - \delta_1 + \delta_{(0,1)}}\Big) \\ \beta_1 &\geq \frac{1}{\delta_0}\Big(\delta_{(0,1)} - \frac{(\delta_0 - \delta_{(0,1)})(\delta_1 - \delta_{(0,1)})}{1 - \delta_0 - \delta_1 + \delta_{(0,1)}}\Big) \end{aligned} \tag{12}$$

We can have the following theorem to hold, with the proof in our technical report [21].

Theorem 4.2: If publishing two terms t_0 and t_1 satisfies conditions in Inequality set 12, then the actual false positive rates of phrase $t_0 t_1$ is statistically larger than those of t_0 and t_1, that is, $fp_{(0,1)} \geq \max(fp_0, fp_1)$.

With this property and Equation 15, we can have the per-phrase false positive rate (e.g. $fp_{(0,1)}$) to be statistically larger than $\max \epsilon_{kk} \geq$ which is further larger than user-configured per-phrase degree $\epsilon_{(0,1)}$.

4.1.3 `ENLP`: A NLP-based Approach

The `MaxE` approach and its extension, essentially based on heuristic, may blindly increase ϵ'_j and excessively incur additional search overhead, leading to sub-optimized performance. We propose a second approach for single term-oriented publication, `ENLP`. The idea is to rigorously model the problem as an optimization problem and solve it to minimize the additional search overhead under the privacy constraints.

4. In ϵ-MPPI, the term frequency refers to the number of matching providers in the network.

To model the problem, we start with a simple two-term case. Consider two terms t_1, t_0 are published separately with ϵ'_0 and ϵ'_1. When publishing term t_0, it is expected that an ϵ'_0 portion of positive owners in the published M' is false positive. Likewise when publishing term t_1, among all positive owners on term t_1, an ϵ'_1 portion is false positive. Because the true positive owners regarding two-term phrase $t_1 t_0$ are those that possess both terms. Thus the true positive rate after publishing two terms $t_1 t_0$ (assuming terms are distributed independently) is $(1-\epsilon'_0)(1-\epsilon'_1)$. We illustrate the intuition in Figure 4, in which the gray area indicates the $(1-\epsilon'_0)(1-\epsilon'_1)$ portion. Given a two-term phrase, say $r_3 = t_1 t_0$, we can formulate the following equation.

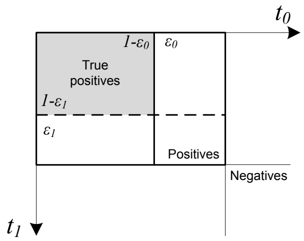


Fig. 4: True positives in single term-oriented publications

$$1 - \epsilon_3 = (1-\epsilon'_0)(1-\epsilon'_1) \tag{13}$$

By generalizing Equation 13 to the multi-term case (with more than 2 terms), one can naturally derive the following,

$$\begin{aligned}
1-\epsilon_0 &\leq (1-\epsilon'_0)^{a_{0,0}} \cdot (1-\epsilon'_1)^{a_{0,1}} \ldots (1-\epsilon'_{n-1})^{a_{0,n-1}} \quad (14)\\
&\ldots\\
1-\epsilon_k &\leq (1-\epsilon'_0)^{a_{k,0}} \cdot (1-\epsilon'_1)^{a_{k,1}} \ldots (1-\epsilon'_{n-1})^{a_{k,n-1}}\\
&\ldots\\
1-\epsilon_{l-1} &\leq (1-\epsilon'_0)^{a_{l-1,0}} \cdot (1-\epsilon'_1)^{a_{l-1,1}} \ldots (1-\epsilon'_{n-1})^{a_{l-1,n-1}}
\end{aligned}$$

Here, recall that $a_{k,j}$ is either 0 or 1; when phrase r_k does not have term t_j, $a_{k,j} = 0$ and item $(1-\epsilon'_{k,j})^a_{k,j} = 1$ which does not contribute to $1 - \epsilon_k$.

Given a number of multi-term searches, we model the additional search cost approximately as below.

$$q_0 \cdot \epsilon'_0 + q_1 \cdot \epsilon'_1 + \cdots + q_{n-1} \cdot \epsilon'_{n-1} \tag{15}$$

Here, q_j for term t_j is the accumulated frequency that term t_j is involved in a search. For instance, if there are 2 searches on phrase $t_1 t_0$ and 3 searches on phrase t_0, then $q_0 = 2+3 = 5$. There are approximately $q_j \cdot \epsilon'_0$ false positive owners contacted for all the searches involving term t_j. Note that in Equation 15, we deliberately omit the potential impact from phrase frequencies since they are sensitive and entails expensive secure computations, as will be discussed (§ 5). In this sense, our goal is to minimized Equation 15 under the privacy constraints as in Inequality set 14.

We formulate an optimization problem by using notations $y_k = -\ln(1-\epsilon_k)$ and $x_j = -\ln(1-\epsilon'_j)$. The problem is stated as below.

$$\begin{aligned}
&\text{Maximize} && q_0 \cdot e^{-x_0} + q_1 \cdot e^{-x_1} + \cdots + q_{n-1} \cdot e^{-x_{n-1}} \quad (16)\\
&\text{Subject to} && a_{0,0} \cdot x_0 + a_{0,1} \cdot x_1 + \ldots + a_{0,n-1} \cdot x_{n-1} \leq y_0\\
& && a_{1,0} \cdot x_0 + a_{1,1} \cdot x_1 + \ldots + a_{1,n-1} \cdot x_{n-1} \leq y_1\\
& && \ldots\\
& && a_{l-1,0} \cdot x_0 + a_{l-1,1} \cdot x_1 + \ldots + a_{l-1,n-1} \cdot x_{n-1} \leq y_{l-1}\\
& && \forall k, j, a_{k,j} = \{0, 1\}
\end{aligned}$$

The problem is a non-linear programming problem (NLP), with linear constraints and a non-linear objective function. Solving this problem can be based on existing solvers (e.g. ILOG's CPLEX or Mathematica). In our implementation, we choose Mathematica's primitive NMAXIMIZE to solve the problem. Note that in our design, the problem does not involve any sensitive variables (e.g. ϵ is non-private by itself) and can be realized by non-secure computations.

4.2 Phrase-wise Publication

The single term-oriented publication may cause a sub-optimized level of privacy preservation due to the ignorance of innate correlations between terms. We consider another general approach, named phrase-wise publication, in which the randomized publication is directly applied at the granularity of multi-term phrases. To be specific, β's are calculated for all private phrases. For phrase r_k with preferred privacy degree ϵ_k, we can calculate the per-phrase $\beta'(r_k)$ in a similar way to the single-term case. The same three policies can apply; for example, if the basic policy is used, the per-phrase $\beta_b(r_k)$ can be calculated using Equation 3.

$$\begin{aligned}
(1-\sigma'_k)\beta'(r_k) &= \sigma'_k \frac{\epsilon_k}{1-\epsilon_k} \quad (17)\\
\Rightarrow \beta'(r_k) &= [(\sigma'^{-1}_k - 1)(\epsilon_k^{-1} - 1)]^{-1}
\end{aligned}$$

Note that σ'_k denotes the frequency for phrase r_k.

While the β computation can be similar to that in the single-term case, the randomized publication process must be changed; because unlike the single-term publication, there could be overlaps between the publishing phrases (different phrases could contain the same term). We propose an iterative approach, named `IBeta`, for the randomized publication of multi-term phrases.

4.2.1 `IBeta`: *An Iterative Approach*

We first formalize the problem that an owner publishes a multi-term phrase (e.g. r_k) with a single probability (i.e. $\beta'(r_k)$). Given phrase r_k, the $\beta'(r_k)$ indicates the probability at which a non-positive owner publishes data as a positive owner. In a multi-term context, an owner is positive when it possesses all terms of the phrase; otherwise, it is non-positive. The publication process first distinguishes between the positive and non-positive cases for an owner, and then applies only for the non-positive owners. For example, consider three phrases in a three-term vocabulary, $r_0 = t_0$, $r_1 = t_1 t_0$ and $r_2 = t_2 t_1$. We can have three per-phrase probabilities, $\beta'(r_0), \beta'(r_1)$ and $\beta'(r_2)$. The intended publication process can be formalized as below.

$$\begin{aligned}
\beta'(r_0): &\quad \cdot\cdot\overline{1} \rightarrow \quad \cdot\cdot 1\\
\beta'(r_1): &\quad \cdot\overline{11} \rightarrow \quad \cdot 11\\
\beta'(r_2): &\quad \overline{11}\cdot \rightarrow \quad 11\cdot
\end{aligned}$$

Here, each $\beta'(r_k)$ is associated with two states: a starting state and an end state. We use the starting state to indicate all possible non-positive cases for a publication. For example, the starting state of the publication with $\beta'(r_1)$ is $\cdot\overline{11}$, which means any owners who do not possess term t_1 and t_0 at the same time. Here $\cdot$ means either 0 or 1, and the line above text means negation. For example, $\overline{11} = \{00, 01, 10\}$.

To publish data with multiple probabilities for overlapping phrases, we propose to use the `IBeta` approach. Algorithm 1 illustrates how the index publication approach iteratively runs, phrase by phrase. Given a series of private phrases $\{r_k\}$, the personal server would retrieve phrase r_k and its corresponding $\beta'(r_k)$ in a topologically sorted order. To be specific, all the private phrases $\{r_k|\forall k\}$ can conceptually form a DAG (directed acyclic graph), in which a node represents a phrase and a directed edge from node r_a to node r_b represents the case that phrase r_b has exactly one term more than phrase r_a. For example, phrase t_2t_1 points to phrase $t_2t_1t_0$. The DAG is topologically sorted and outputs all the nodes r_k's with the corresponding $\beta'(r_k)$'s in order. For phrase r_k and probability $\beta'(r_k)$, the owner checks whether its current local vector matches with the $\beta'(r_k)$'s starting state. If matched, it proceeds to publish the current local vector accordingly. This process completes until it goes through all the $\beta'(r_k)$'s.

Algorithm 1 iterative-publish(owner p_i, set $\{\beta'(r_k)\}$)

1: **for all** $k \in [0, l-1]$ **do** $\triangleright$ $\beta'(r_k)$ is topologically sorted
2: **if** match(cur_memvec, getStartingState(r_k)) **then** $\triangleright$ cur_memvec is the current membership vector
3: cur_memvec$\leftarrow$publish(cur_memvec, $\beta'(r_k)$)
4: **end if**
5: **end for**

Example: We follow the previous example to illustrate the iterative process. The three phrases, that is, $r_0 = t_0, r_1 = t_1t_0, r_2 = t_2t_1$, can be represented by a membership vector $\{a(0), a(1), a(2)\} = \{001, 011, 110\}$. Topologically sorting the three phrases results in a possible ordering, r_0, r_1, r_2 or $001, 011, 110$. Consider an owner who possesses only term t_1 needs to publish its local vector, 010. After receiving the term vector $\{r_0 : \beta'(r_0), r_1 : \beta'(r_1), r_2 : \beta'(r_2)\}$, the owner considers each publishing probability β' in the topological order. In the first iteration, it considers phrase r_0 and probability $\beta'(r_0)$; since the owner's vector 010 matches the starting state of r_0, that is, $\cdot\cdot\overline{1}$, it then attempts to flip the value 0 at t_0's position to 1 with probability $\beta'(r_0)$. Assume it has the luck and successfully flips the value, which results in the vector changed to 011 (from 010). The second iteration considers $\beta'(r_1)$ whose starting state is $\cdot\overline{11}$. It does not match the current vector, 011. Then it moves on to the third phrase, that is, r_2 with probability $\beta'(r_2)$. Likewise, owner p_1 can determine that the current vector, 011, matches the starting state, $\overline{11}\cdot$. It then follows the protocol to flip vector 011 (potentially to 111) based on probability $\beta'(r_2)$. Assume this time it does not have the luck and the final vector is 011.

Security property: Given the publishing process in `IBeta`, multiple iterations could exist there to publish just one term. For example, two phrases r_0 and r_1 both have term t and `IBeta` will use two iterations for publishing t. The publishing probabilities to flip the negative incident binary for the two iterations are $\beta(r_0)$ and $\beta(r_1)$, then the final publishing probability for term t would be:

$$\begin{aligned} \beta &= \beta(r_0) + (1 - \beta(r_0)) \cdot \beta(r_1) \\ &= \beta(r_0) + \beta(r_1) - \beta(r_0) \cdot \beta(r_1) \\ &\geq \max(\beta(r_0), \beta(r_1)) \end{aligned} \tag{18}$$

Note the derivation is based on the fact that the second iteration is only effective when the first iteration fails to flip the binary. The property guarantees that the publication of one term can meet the privacy requirements of all phrases covering the term.

5 SECURE ϵ-MPPI CONSTRUCTION

The previous two sections describe the computation model of publishing ϵ-MPPI, and in this section, we discuss the realization of such computation in a secure way yet without assuming any mutual trusts between servers. We first introduce a general secure computation framework primarily for single-term publication in ϵ-MPPI and then describe the specific extension and optimization for the multi-term publication.

The ϵ-MPPI construction takes input of the possession data (or local vectors) sensitive to each server, and outputs the obscured possession data (with proper amount of false positives) to the ϵ-MPPI server. The construction of ϵ-MPPI is then a three-stage process, as shown in Figure 5; the first two stages correspond to the multi-source analytical computation described before. The first stage runs a protocol called `AggSharedSum`, which computes the sum of all possession information. The output of `AggSharedSum` is frequencies for terms and phrases. Due to the common-term vulnerability discovered by our previous work [22], it calls for protection of the frequency information. We thus design the `AggSharedSum` protocol to distribute the shares of sensitive frequency information to c servers which act as c coordinators for the subsequent stages. These c coordinators represent c disjoint groups of servers in the entire network, and are assumed not to collude with each other. The next stage is a generic MPC (multi-party computation) applied among those non-colluding c coordinators. By reducing the expensive MPC to be among c servers, we can achieve efficiency and make it possible for large-scale computations. The MPC outputs β_k's and feeds them to the third stage – the randomized publication. Here, the computed β_k's are safe to be public and randomized publication is a parallel computation process which happen independently on all the servers. The details of this general computing framework can be found in our previous work [22].

Our ϵ-MPPI construction is implemented as an extension to the above computation framework. Such an extension is realized at different stages depending on protocols, as shown by red rectangles in Figure 5. For the single term-oriented publication (including `MaxE` and `ENLP`), it is

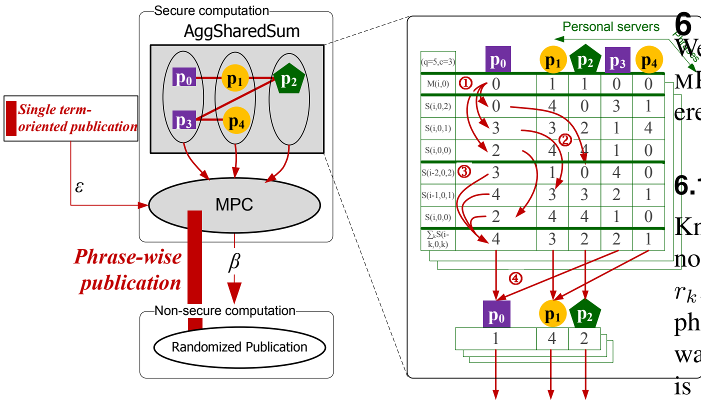


Fig. 5: Distributed computations in ϵ-MPPI construction

essentially a process to convert the per-phrase privacy degree (ϵ_k) to the per-term privacy degree (ϵ'_j). Such conversion is realized as a non-secure computation before the `AggSharedSum`. Because the goal of single-term oriented protocol is to reuse the single-term publication, `AggSharedSum` is configured to calculate frequencies for terms. For phrase-wise publication (i.e. the `IBeta` approach) the first two stages are configured for computing per-phrase frequencies, and the randomized publication runs the iterative `IBeta` process to publish the local vectors.

5.1 Protocol Analysis

Complexity analysis: The complexity of ϵ-MPPI construction is linear to the number of servers m (when the parameter c is a constant). Because in our construction, particularly in the `AggSharedSum` protocol, there are fixed number of rounds, with each server in each round sending/receiving constant messages (that is, c messages), thus $O(m * c)$ messages in total.

Security property: We analyze the security property of the `AggSharedSum` protocol. Basically `AggSharedSum` is by itself a distributed secret sharing protocol; the output c shares can protect information secrecy as described in Theorem 5.1. The proof can be found in our technical report [21].

Theorem 5.1: The `AggSharedSum` protocol is a (c, c) secret sharing scheme of the sum of its inputs. Specifically, `AggSharedSum` takes m ($m \gg c$) inputs, $v_i = M(i, j)$, and produces c outputs, $\{s_{ii}, \forall ii \in [0, c-1]\}$. The outputs are shares of the sum of the inputs with the following properties.

- *Recoverability*: Given c output shares, the secret value (i.e. the sum) can be easily reconstructed.
- *Secrecy*: Given any $c-1$ or fewer output shares, one can learn nothing about the secret value of the sum, in the sense that the conditional distribution given the known shares is identical to the prior distribution. Formally, we have the equation below.

$$\forall x \in \mathbb{Z}_q, Pr(\sum_{\forall i} v_i = x) = Pr(\sum_{\forall i} v_i = x | V \subset \{s_{ii}\})$$

where V is any proper subset of $\{s_{ii}\}$.

6 Privacy Analysis

We analyze the privacy preservation of our proposed ϵ-MPPI against attackers of different capabilities as considered in our threat model.

6.1 A: Knowing the public PPI (M')

Knowing the published ϵ-MPPI data M', the attacker can not have a confidence higher than ϵ_k on a search of phrase r_k. We consider the single-term and multi-term cases. For phrases of a single term t_j, we compute $\beta(t_j)$ in such a way that the false-positive rate in the published ϵ-MPPI is larger than ϵ_j, thus achieving ϵ-PHRASE-PRIVACY. For multi-term phrase publication, we enforce the property that the per-phrase false-positive rate is always higher than the per-term false positive rate as in the extended `MaxE` and `IBeta`; recall Theorem 4.2 and Equation 19. Since we set per-term false positive rate to be higher than any ϵ_k where phrase r_k covers the term, we can ensure that the phrase publication also achieves ϵ-PHRASE-PRIVACY. It is noteworthy that our ϵ-MPPI is fully resistant to repeated attacks against the same term or phrase over time, because our index is static; once published, the index and protection level stays the same and unchanged.

6.2 B: Knowing some ground truth (M)

We follow our attack model (§ 2.2) and consider both concrete attack cases, B.1) and B.2).

6.2.1 Case B.1

In this case, an attacker is able to verify accessible part in the search result from ϵ-MPPI. Specifically, given the search result on phrase r_k, the attacker can see two types of search results: 1. accessible true positive owners, who possess accessible documents which are relevant to r_k, 2. uncertain owners, on whom all the documents accessible to the searcher are irrelevant to r_k. There are two actual situations that could happen under case 2: 2.1, uncertain true positive owners who actually have documents relevant to r_k, but such documents are not accessible to the searcher, 2.2, uncertain false positive owners, who do not have any documents relevant to r_k. The searcher can not distinguish the two situations (i.e. 2.1 and 2.2), and this property allows ϵ-MPPI to achieve even higher level of privacy preservation, as can be seen in the following example.

Example: Consider a search result includes 3 true positive servers and 2 false positive servers. We assume among the 3 true positive servers one is inaccessible to the owner, that is, case 2.1. If the searcher does not verify the result from accessible servers (i.e. attack A), all she can see is 5 servers in the result, leading to the false positive rate being $\frac{2}{5}$. If the searcher verifies the result by accessible servers (i.e. attack A and B.1), she can see two accessible true positives, and the other three being uncertain servers without distinguishing actual situations (i.e. being case 2.1 or 2.2). Therefore, the only true positive is hidden with the two false positives, leading to a higher false positive rate $\frac{2}{3} > \frac{2}{5}$.

TABLE 3: An attacker's view with accessible ground truth M

	p_0	p_1	p_2	p_3	p_4
Attacker's view	1	?	1	?	?
Ground truth	1	1	1	0	0

6.2.2 Case B.2

In this case, the attacker can collude with server owners, and thus is able to access *all* documents on the colluding server. The attacker can distinguish case 2.1 and 2.2 (as described above) if a result server is in collusion. Then ϵ-MPPI loses control of achieving quantitative privacy degree, which now also depends on the number of colluding servers.

An effective attack strategy is that for a false positive phrase r_k on server p_i, the attacker can target on phrase r_k on owners p_j other than p_i and with $M'(j,k) = 1$. This strategy improves the attack success ratio because it eliminates false positives through colluding owners. This attack makes the privacy protection dependent on the attacker's ability to form collusion, thus leaving our ϵ-MPPI's privacy degree at FIXEDPROTECTION.

Essentially in the attack, we use both M and M' information, that is, combining attack $B.2$ and A. The attacker can further improve the success ratio by exploiting phrase-frequency information revealed in ϵ-MPPI construction (i.e. attack A, B.2, C). For instance, the frequency and configured sensitivity ϵ allow one to deduce the total amount of false positives. If it happens that all the false positives are on the colluding servers, the attacker can be 100% certain that the true positives are on the non-colluding servers, leaving privacy definitely leaked.

Although ϵ-MPPI can not provide a quantitative guarantee for the B.2 attack family, in practice the assumed attack conditions rarely hold – the probability to form collusion decreases exponentially with the number of colluding servers. To prevent rare phrases with limited false positives become vulnerable, we bound the minimal number of false positives, that is, all phrases' false positives must be larger than a pre-configured threshold. This policy minimizes the possibility for a colluding attack to succeed.

6.3 C: Knowing index construction

ϵ-MPPI construction consists of three main stages: the `AggSharedSum`, MPC and the non-secure stage for randomized publication. The information involved in the non-secure computation is fully exposed without any protection. In the following, we first analyze the first two secure computation protocols and then the non-secure protocol.

For the secure-sum protocol, we consider the attacker can collude with servers in the hope of gaining useful information revealed in the ϵ-MPPI construction process and exploiting it to her advantage. We use the semi-honest model for the servers, as used in other MPC protocols [15]. In particular, the `AggSharedSum` protocol can guarantee: 1) $(c-1)$-secrecy of the input: Based on the fact that each secret input is decomposed to c shares and distributed to $c-1$ peer servers, it is easy to understand the $c-1$ input secrecy. With less than c servers in collusion, none of any private input can be learned by any servers other than its owner. 2) c-secrecy of the output: Based on Theorem 5.1, the phrase/term frequency can only be reconstructed when all c shares are used. With less than c shares, one can learn nothing. The generic MPC technique can provide information confidentiality against up to c colluding servers [16]. Overall, as long as the attacker does not collude with more than $c-1$ servers in the network, ϵ-MPPI can protect privacy and guarantee information-theoretic security in confidentiality.

The non-secure randomized publication takes β and phrase frequencies as input. Here, β, calculated from the second stage, does not carry any private information, and is thus safe to be released to the untrusted servers. In addition, only frequencies of non-common phrases are released while those of common phrases which are sensitive are obscured properly by the second-stage MPC process, thus privacy preserved.

6.4 Co-occurrence attack

A co-occurrence attack is to exploit the co-occurrence statistics of multiple terms in order to identify false positives, and further to identify the vulnerable true positives. Considering a concrete scenario where 1) two terms are with zero co-occurrence statistics, and 2) those two terms do co-appear on one server (observed from public PPI data M'). Based on the two facts, the attacker can be sure that *at least one of the two terms is false positive on the server*. This essentially discloses linkage between two published terms; if one can know (through certain channel) that one term (e.g. t_0) is true positive, then she can be sure that the other term (e.g. t_1) must be false positive too; this information can be used to further identify the case of t_1 on other servers.

Assume t_0 has term frequency $\frac{1}{m}$; that is, t_0 appears only once on all servers. This information can be disclosed through attack C. Also assume M' reveals (as in attack A) that there are two servers where t_0 appear to be (true or false) positive. Given that t_0's co-occurrence with t_1 is zero and they do appear on one server p_0, if the attacker can access and know that t_1 is true positive on p_0 (e.g. as in attack B.1), she can then infer that t_1 must be true positive on the other sever p_1. By this means, private possession information of t_1 by p_1 is disclosed.

In the case of co-occurrence attacks, ϵ-MPPI provides privacy protection at degree FIXEDPROTECTION. In practice, it is a corner case for the co-occurrence attack to succeed, as it requires different information obtained through combining multiple attack types (that is, Attacks A, $B.1$ and C) and relies on certain term distribution to happen (that is, zero co-occurrence and small per-term frequency). We anticipate the probability for the attack conditions to be true is low, thus the amount of vulnerable phrases/terms is small. Notice that the co-occurrence attack works only for rare term/phrase with small frequency. In our protocol (as mentioned before), we bound the minimal amount of false positives, so that it essentially eliminates the co-occurrence vulnerability on rare terms/phrases.

7 Experiments

To evaluate the proposed ϵ-MPPI, we conducted two sets of experiments: The first set evaluates the effectiveness of ϵ-MPPI in quantitative privacy protection. The second set evaluates the performance of our ϵ-MPPI construction protocols. For realistic performance study, we have implemented a functioning prototype for ϵ-MPPI construction.

7.1 Quality of Privacy Preservation

Experimental setup: To simulate the information network, we used a distributed document dataset [23] of 2,500 - 25,000 small digital libraries, which is further derived from NIST's publicly available TREC-WT10g dataset [24]. To be specific, this dataset defines two tables, a "collection" table and a query table. The collection table maintains the mapping from the documents (each with a unique ID defined in the TREC-WT10g dataset) to the collections. Each collection is generated based on the document URLs in the WT10g dataset and simulates a document set in a digital library (as in the original work [23]). In our setting, a collection is used to simulate a personal server. The query table maintains a list of known item queries; for each query, it keeps the mapping from a multi-term phrase to a document. The query table and the collection table collectively emulate matrix M. Table 4 summarizes the multi-term queries available from the dataset with their lengths and frequencies.

TABLE 4: Multi-term query distribution

NumberOfTerms	NumberOfQueries	Percentage
1	114,404	6.91%
2	658,870	39.79%
3	482,829	29.16%
4	375,201	22.66%
5	20,164	1.22%
6	4,297	0.26%

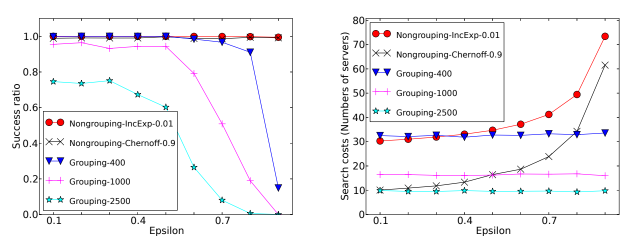
The dataset does not have privacy metric for the query phrase. In our experiment, we randomly generate privacy degrees (ϵ) in interval $[0, 1]$.

To evaluate the effectiveness of privacy preservation, we use two metrics: the publication success rate (p_p) and extra search cost. Recall that for a phrase, the success rate measures the likelihood that the published PPI meets the privacy requirement regarding the phrase. Here, we consider the average success rate for all the private phrases. We also use the extra search overhead, which is the number of false positive owners that are "excessive" for privacy preservation. For example, if 5 false positive servers suffice to preserve the privacy and the actual constructed PPI contains 7 false positives, then the excessive number is $7 - 5 = 2$. This amount of false positive owners are excessive and unnecessary as they do not contribute to achieving the desired privacy preservation.

7.1.1 Non-grouping and Grouping PPI's

Most existing PPI work is based on a grouping abstraction; it clusters owners or personal servers into privacy groups in resemblance to k-anonymity [25], so that different owners in the same privacy group can not be distinguished from each other. In this regards, our ϵ-MPPI is non-grouping in nature. The experiments compare the non-grouping and grouping approaches for PPI's. We still use the metrics of the success rate and extra search costs. In the experiment, grouping PPI's are tested under different group sizes. Given a network of owners, we use the number of groups to control the average group size. We test grouping PPI with the `Chernoff` and `IncExp` policies under the default setting. We set $m = 10,000$. We run the experiments 20 times and report the averaged results.

We show the result with changing privacy degrees (ϵ), which demonstrates that ϵ-MPPI can perform equally well for both sensitive and non-sensitive terms (i.e. with large and small values of ϵ). In Figure 6 we report the success rate and the search costs under different privacy preservation degrees. It can be clearly seen that the non-grouping ϵ-MPPI can achieve much better quality of privacy preservation than the grouping PPI. While the grouping PPI's with different configurations (in the group number) achieve constant search costs at the expenses of unstable privacy preservation levels (as in Figure 6b), the non-grouping ϵ-MPPI can achieve high success rate that meets the requirement (i.e. γ), in spite of different values of ϵ or different frequencies. In particular, when ϵ grows large in Figure 6a, the success rate of grouping PPI's quickly degrades to 0, rendering its privacy preservation quality unacceptable. This stems from the grouping PPI's design that treats different terms and phrases in the same way. This set of experiments shows that the privacy degree of non-grouping ϵ-MPPI can be effectively tuned in practice, implying the ease of configuration and more privacy control exposed to applications.

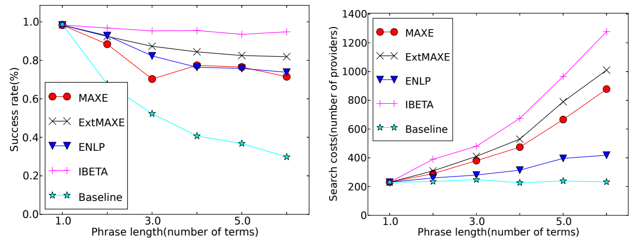


(a) Success rate p_p (b) Search costs

Fig. 6: Comparing non-grouping and grouping PPIs under ϵ

7.1.2 Effectiveness of preserving multi-term privacy

To evaluate the effectiveness of the multi-term privacy preservation, we conducted experiments based on the proposed publication approaches, including `MaxE`, `ENLP` and `IBeta`. For comparison, we use a very straightforward approach as the baseline, since existing PPI's do not particularly address the multi-term privacy. In our baseline, we ignore all the privacy constraints on multi-term phrases but



(a) Success rate (b) Extra search cost

Fig. 7: Privacy preservation of multi-term phrases

rather consider only those related to single-term phrases. For fair comparison, we used the `Chernoff` policy for all the multi-term publication approaches.

For each setting, we have run the experiments more than 30 times and report the average results. Figure 7 summarizes our experimental results. In terms of the success rate, Figure 7a shows the result of different approaches with phrases of different lengths (i.e. number of terms in a phrase). As the phrase length increases, the baseline approach's success rate drops significantly, which is expected. Because the baseline approach does not take into account the privacy configuration of multi-term phrases. By contrast, all our proposed approaches are more stable with the changing phrase length. The `IBeta` approach achieves the best success rate among all three approaches; it is always close to the ideal case, 100%. This is due to that `IBeta` considers the case of correlated terms and accordingly preserves the multi-term privacy. By comparison, the other two approaches, `MaxE` and `ENLP`, fluctuate and depart from the ideal case. The main reason behind is that their success rate greatly depends on the phrase and document distribution as they assume independent term distribution in the computation models. If there is not very strong correlation between servers, these two approaches can achieve relatively high success rates. Figure 7b shows the result for extra search cost. Despite the baseline approach which performs worse than others, the `ENLP` approach is the best; because it models the search overhead and optimizes it using our NLP-based technique. As expected, the extra search costs increase as the phrase length grows.

7.2 Performance of Index Construction

Experimental setup: We evaluated the performance of our distributed protocol for secure ϵ-MPPI construction. We implemented a functioning prototype for a realistic performance study. The MPC is implemented by FairplayMP [16]; the computation is realized in SFDL, a secure function definition language exposed by FairplayMP, and is compiled by the FairplayMP runtime to Java code, which embodies the generated circuit for secure computations. We implemented the `AggSharedSum` protocol in Java. In particular, we use a third-party library named Netty [26] for network communications and Google's Protocol Buffer [27] for object serialization. To solve the NLP problem as in the `ENLP` approach we use Mathematica's function NMAXIMIZE. We conducted experiments on a number of machines in Emulab [28], [29], each equipped with a 2.4 GHz 64-bit Quad Core Xeon processor and 12 GB RAM. In the experiment, different numbers of machines are tested, from 3 to 9. For each experiment, the protocol is compiled to and runs on the same number of parties; each party is mapped to one dedicated physical machine. In the experiment we configured $c = 3$.

We first tested the performance of different index construction approaches, including `MaxE`, `ENLP` and `IBeta`. We have measured the execution time of our implemented ϵ-MPPI construction protocol in a three-node network. We varied the number of terms per phrase from 1 to 6 as in Table 4. The execution time is reported in Figure 8. Basically, the `IBeta` approach is the most consuming and the time increases exponentially with the number of terms (or linearly with the number of phrases). This is due to that the `IBeta` approach makes per-phrase use of generic MPC, which is more expensive than the per-term use (there are more phrases than terms). In the `ENLP` approach, the execution time also increases exponentially, largely caused by the NLP computation. However, as the NLP is carried out by a non-secure computation on a centralized party, it is much less dominant in the overall computation, which makes the `ENLP` approach more efficient than the `IBeta` approach. The most efficient approach is `MaxE`, whose execution time increases slowly with the number of terms. Because the per-phrase computation is mainly caused by Equation 4, which is lightweight.

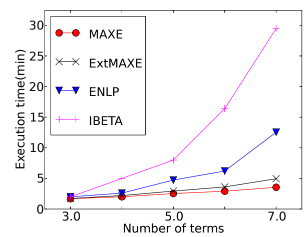
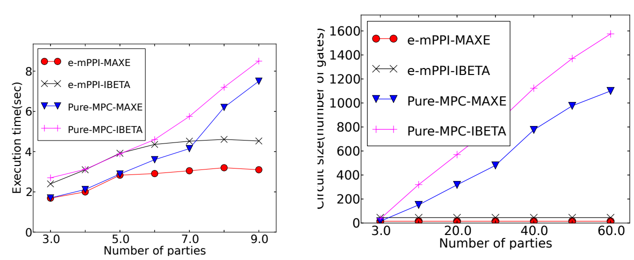


Fig. 8: Performance of different index construction approaches



(a) Execution time (b) Circuit size (number of gates)

Fig. 9: Scalability of index construction protocol

To verify our design standpoint that MPC is expensive, we compare our approach with a baseline approach based on the pure use of MPC. The pure MPC approach the generic MPC on all servers and for all computations of β, without the MPC-reduction technique used in `AggSharedSum`. In addition to the pure MPC approach, we implemented `MaxE` and `IBeta`. The metric used in the experiment is an end-to-end execution time, which measures the time duration from when the protocol starts to run to when the last machine in our cluster reports to finish. The result is shown in Figure 9a. It can be seen that the pure

MPC approach generally incurs longer execution time than the ϵ-MPPI approach: As the network grows large, while the execution time of pure MPC approach increases super-linearly, that of the ϵ-MPPI approaches increases slowly. This difference is due to that the MPC computation in our ϵ-MPPI approach is fixed to c parties where $c \ll m$ so that the execution time almost does not change with the number of servers.

For experiments with a larger number of parties or servers, we use the compiled circuit size of the protocol as the (emulated) metric. The circuit size determines the preprocessing time (or the compile time) of the protocol and its execution time[5] in a real run. By this means, we can show the scalability result of up to 60 parties as in Figure 9b. The similar performance improvement can be observed except that the circuit size grows linear with the number of parties involved.

8 Related Work

8.1 Secure Indexing on Untrusted Servers

PPI is designed to index access controlled contents scattered across multiple personal servers. Since it is hosted on an untrusted server, the PPI aims at preserving the content privacy of all participant servers. Inspired by the privacy definition of k-anonymity [25], existing PPI work [9], [11], [10] follows the *grouping-based* construction; it organizes servers into disjoint privacy groups, such that servers from the same group are indistinguishable to the searchers. To construct such an index, existing approaches [9], [11], [2] assume servers are willing to disclose their private local indexes, which is unfortunately an unrealistic assumption in a network lack of mutual trusts between servers. SS-PPI [10] is proposed with resistance against colluding attacks. While most existing grouping PPI utilizes a randomized approach to form groups, its weakness is studied in SS-PPI but without a viable solution. Though the group size can be used to configure grouping-based PPI, it lacks per-term concerns and quantitative privacy guarantees. Moreover, organizing servers in groups usually leads to query broadcasting (e.g, with positive servers scattered in all groups), rendering the search inefficient. By contrast, ϵ-MPPI and our previous work [22] are based on a brand new PPI abstraction without the use of grouping; it can provide quantitative privacy control on a per-phrase basis. Unlike the PPI scheme, which is designed to be hosted on untrusted servers, Zerber [30] assumes partial trusts on the hosting server; Zerber decomposes the index structure along with authentication keys into shares and stores them on an array of hosting servers which are assumed not to collude. This scheme however comes with non-negligible performance penalty for data and query serving; instead of contacting one PPI server, Zerber has to contact multiple servers in order to perform a meaningful search, which deteriorates the search performance. In addition, Zerber indexes at the document level and assumes a fixed and small number of servers in the network, while our ϵ-MPPI indexes at the provider/server level and assumes a large number of servers in the network. Our previous work [22] focuses on the single-term phrase protection.

Another related area is data indexing in P2P networks [31], [32], [33]. Those P2P indices are built on top of Distributed Hash Tables (or DHT) and distributed to multiple peers/nodes in DHT. While our ϵ-MPPI is currently assumed to be served on a centralized entity, it is straightforward to extend ϵ-MPPI's architecture to P2P index serving; ϵ-MPPI can be served as a P2P index if a DHT structure can be imposed on the information network which achieves better load balancing and scalability.

8.2 Privacy Definitions for Anonymization

Publishing public-use data about individuals without revealing sensitive information has received a lot of research attentions in the last decade. Various privacy definitions have been proposed and gained popularity, including k-anonymity [25], l-diversity [34], and differential privacy [35]. In particular, in a k-anonymized dataset, each record is indistinguishable from at least $k-1$ other records. This idea is applied in the PPI setting; most existing PPI uses the grouping notion to make servers k-anonymized in the public-use PPI. We propose a non-grouping ϵ-MPPI which demonstrates the promise for better quality of privacy preservation. ϵ-MPPI utilizes a new privacy definition, ϵ-PHRASE-PRIVACY, to particularly address the privacy with multi-term document searches. The most relevant privacy definition to our ϵ-PHRASE-PRIVACY degree is r-confidentiality [30] which also addresses the privacy preservation of a PPI system for public use. However, r-confidentiality does not particularly consider the case of multi-term phrases.

9 Conclusion

In this paper, we propose ϵ-MPPI for multi-term phrase publication with quantitative privacy control in emerging information networks. We propose several practical approaches for the secure construction of an ϵ-MPPI system in an environment without mutual trusts, while being able to provide the multi-term privacy. For practical performance of secure computations, we propose an MPC-reduction technique based on the efficient use of secret sharing schemes. We also discovered a common-term vulnerability and proposed a term-mixing solution. Through both simulation-based and real experiments, we show the advantage of ϵ-MPPI in terms of privacy preservation quality and construction efficiency.

10 Acknowledgment

This work was performed under a partial support by the National Science Foundation under Grants IIS-0905493, CNS-1115375, IIP-1230740 and Intel ISTC on cloud computing.

5. Detailed correlation between circuit size and execution time can be found in the experiment study in FairplayMP [16].

REFERENCES

[1] R. J. B. Jr., R. Agrawal, D. Gruhl, and A. Somani, "Youserv: a web-hosting and content sharing tool for the masses," in *WWW*, 2002, pp. 345–354.

[2] M. Bawa, R. J. B. Jr., S. Rajagopalan, and E. J. Shekita, "Make it fresh, make it quick: searching a network of personal webservers," in *WWW*, 2003, pp. 577–586.

[3] "Diaspora: https://joindiaspora.com/."

[4] "Status, http://status.net."

[5] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *SIGCOMM*, 2009, pp. 135–146.

[6] H. Löhr, A.-R. Sadeghi, and M. Winandy, "Securing the e-health cloud," in *IHI*, 2010, pp. 220–229.

[7] "Nhin direct, http://directproject.org/."

[8] R. Geambasu, M. Balazinska, S. D. Gribble, and H. M. Levy, "Homeviews: peer-to-peer middleware for personal data sharing applications," in *SIGMOD Conference*, 2007, pp. 235–246.

[9] M. Bawa, R. J. B. Jr., and R. Agrawal, "Privacy-preserving indexing of documents on the network," in *VLDB*, 2003, pp. 922–933.

[10] Y. Tang, T. Wang, and L. Liu, "Privacy preserving indexing for ehealth information networks," in *CIKM*, 2011, pp. 905–914.

[11] M. Bawa, R. J. Bayardo, Jr, R. Agrawal, and J. Vaidya, "Privacy-preserving indexing of documents on the network," *The VLDB Journal*, vol. 18, no. 4, 2009.

[12] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: protecting confidentiality with encrypted query processing," in *SOSP*, 2011, pp. 85–100.

[13] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC*, 2009, pp. 169–178.

[14] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," in *INFOCOM*. IEEE, 2011, pp. 829–837.

[15] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay - secure two-party computation system," in *USENIX Security Symposium*, 2004, pp. 287–302.

[16] A. Ben-David, N. Nisan, and B. Pinkas, "Fairplaymp: a system for secure multi-party computation," in *ACM Conference on Computer and Communications Security*, 2008, pp. 257–266.

[17] W. Henecka, S. Kögl, A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, "Tasty: tool for automating secure two-party computations," in *ACM CCS*, 2010, pp. 451–462.

[18] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen, "Asynchronous multiparty computation: Theory and implementation," in *Public Key Cryptography*, 2009, pp. 160–179.

[19] A. Narayan and A. Haeberlen, "DJoin: differentially private join queries over distributed databases," in *OSDI*, Oct. 2012.

[20] J. Hsu, M. Gaboardi, A. Haeberlen, S. Khanna, A. Narayan, B. C. Pierce, and A. Roth, "Differential privacy: An economic method for choosing epsilon," *CoRR*, vol. abs/1402.3329, 2014. [Online]. Available: http://arxiv.org/abs/1402.3329

[21] Y. Tang and L. Liu, "Multi-keyword privacy-preserving search in information networks," *Technical Report 2014, http://tristartom.github.io/docs/tr-mppi.pdf*.

[22] Y. Tang, L. Liu, A. Iyengar, K. Lee, and Q. Zhang, "e-ppi: Locator service in information networks with personalized privacy preservation," in *IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014, Madrid, Spain, June 30 - July 3, 2014*, 2014, pp. 186–197. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2014.27

[23] J. Lu and J. P. Callan, "Content-based retrieval in hybrid peer-to-peer networks," in *CIKM*, 2003, pp. 199–206.

[24] D. Hawking, "Overview of the trec-9 web track," in *TREC*, 2000.

[25] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[26] "Netty: http://netty.io."

[27] "Protobuf: http://code.google.com/p/protobuf/."

[28] "http://www.emulab.net/."

[29] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *OSDI*, 2002.

[30] S. Zerr, E. Demidova, D. Olmedilla, W. Nejdl, M. Winslett, and S. Mitra, "Zerber: r-confidential indexing for distributed documents," in *EDBT*, 2008, pp. 287–298.

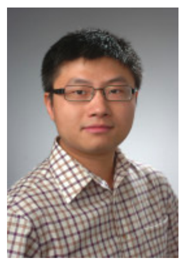
[31] Y. Tang and S. Zhou, "LHT: A low-maintenance indexing scheme over dhts," in *28th IEEE International Conference on Distributed Computing Systems (ICDCS 2008), 17-20 June 2008, Beijing, China*, 2008, pp. 141–151. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2008.61

[32] Y. Tang, J. Xu, S. Zhou, and W. Lee, "m-light: Indexing multi-dimensional data over dhts," in *29th IEEE International Conference on Distributed Computing Systems (ICDCS 2009), 22-26 June 2009, Montreal, Québec, Canada*, 2009, pp. 191–198. [Online]. Available: http://dx.doi.org/10.1109/ICDCS.2009.30

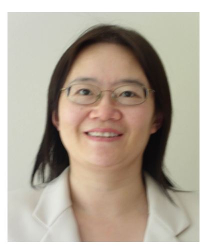
[33] Y. Tang, S. Zhou, and J. Xu, "LIGHT: A query-efficient yet low-maintenance indexing scheme over dhts," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 1, pp. 59–75, 2010. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2009.47

[34] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," in *ICDE*, L. Liu, A. Reuter, K.-Y. Whang, and J. Zhang, Eds. IEEE Computer Society, 2006, p. 24.

[35] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, ser. Lecture Notes in Computer Science, S. Halevi and T. Rabin, Eds., vol. 3876. Springer, 2006, pp. 265–284.



Yuzhe Tang is an Assistant Professor in the Department of EECS at Syracuse University. His research is broadly on distributed systems and security, with current focus on applying secure multi-party computations and building secure key-value stores. Dr. Tang obtained his Ph.D. degree in the College of Computing at Georgia Institute of Technology in 2014. Dr. Tang has his MSc and BSc degrees in Computer Science and Engineering at Fudan University, Shanghai, China. Dr. Tang has published first-author papers in ICDCS, ACSAC, CIKM, TKDE, TPDS. Dr. Tang serves in technical program committees in venues such as ICDCS. He is a recipient of the Best Paper award in IEEE Cloud 2012, and the Best Paper nomination in CCGrid 2015.



Ling Liu is a Professor in the School of Computer Science at Georgia Institute of Technology. She directs the research programs in Distributed Data Intensive Systems Lab (DiSL), examining various aspects of large scale data intensive systems. Prof. Ling Liu is an internationally recognized expert in the areas of Database Systems, Distributed Computing, Internet Data Management, and Service oriented computing. Prof. Liu has published over 300 international journal and conference articles and is a recipient of the best paper award from a number of top venues, including ICDCS 2003, WWW 2004, 2005 Pat Goldberg Memorial Best Paper Award, IEEE Cloud 2012, IEEE ICWS 2013. Prof. Liu is an elected IEEE Fellow in 2014 and a recipient of IEEE Computer Society Technical Achievement Award in 2012 and an Outstanding Doctoral Thesis Advisor award from Georgia Institute of Technology. In addition to services as general chair and PC chairs of numerous IEEE and ACM conferences in data engineering, very large databases and distributed computing fields, Prof. Liu has served on editorial board of over a dozen international journals. Currently Prof. Liu is the editor in chief of IEEE Transactions on Service Computing, and serves on the editorial board of half dozen international journals, including ACM Transactions on Internet Technology (TOIT), ACM Transactions on Web (TWEB), Distributed and Parallel Databases (Springer), Journal of Parallel and Distributed Computing (JPDC). Dr. Liu's current research is primarily sponsored by NSF, IBM, and Intel.