ϵ-PPI: Locator Service in Information Networks with Personalized Privacy Preservation

Yuzhe Tang [§] Ling Liu [†] Arun Iyengar [‡] Kisung Lee [†] Qi Zhang [†]

[§]*Department of EECS, Syracuse University, Syracuse, NY, USA*

[†]*College of Computing, Georgia Institute of Technology, Atlanta, GA, USA*

[‡]*IBM T.J. Watson Research Center, Yorktown Heights, NY, USA*

Abstract

In emerging information networks, having a privacy preserving index (or PPI) is critically important for locating information of interest for data sharing across autonomous providers while preserving privacy. An understudied problem for PPI techniques is how to provide controllable privacy preservation, given the innate difference of privacy concerns regarding different data owners. In this paper we present a personalized privacy preserving index, coined ϵ-PPI, which guarantees quantitative privacy preservation differentiated by personal identities. We devise a new common-identity attack that breaks existing PPI's and propose an identity-mixing protocol against the attack in ϵ-PPI. The proposed ϵ-PPI construction protocol is the first without any trusted third party and/or trust relationships between providers. We have implemented our ϵ-PPI construction protocol by using generic MPC techniques (secure multi-party computation) and optimized the performance to a practical level by minimizing the expensive MPC part.

I. Introduction

In information networks, autonomous service providers store private personal records on behalf of individual owners and enable information sharing under strict enforcement of access control rules. Such information networks have the following salient features: 1) Providers, each under a different administrative domain, do not mutually trust each other; 2) Providers have the responsibility of protecting owners' privacy.

An example of the information network is the emerging HIE or Healthcare Information Exchange systems (e.g. NHIN [1], GaHIN [2] and CommonWell [3]), in which patients delegate their personal medical records to the hospitals that they visited and hospitals form a nation-wide (or state-wide) network to share information. Specifically, different hospitals may compete for the same customer base (i.e. patients) and have conflicting economic interests, which renders it difficult to build full trust relationships between them. Hospitals are responsible for protecting patient privacy, as regulated by Federal laws (e.g. HiPAA [4]). Other examples of the multi-domain information networks include cross-university online course management systems (e.g. Coursera [5] and StudIP [6]), distributed social networks (e.g. Diaspora [7] and Twister [8]) and others.

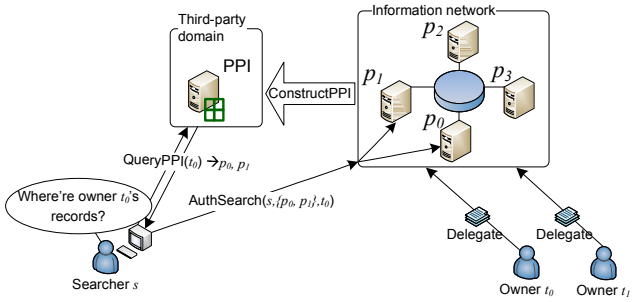


Fig. 1: The system of PPI and information network

Information sharing is crucial for various applications in information networks. In the HIE case, for example, when a patient who is unconscious is sent to a hospital, information sharing between multiple hospitals can help the doctor retrieve the patient's medical history for immediate and accurate medical treatment. To establish information-sharing sessions, Record Locator Service [9], [10] is a standard procedure in the existing HIE systems; it provides the ability to identify where a patient's records are located based upon her identity, and is used as the first step towards sharing information between the searcher and the patient's hospital of interest. Internally, a locator service maintains meta-data regarding the patients' medical history (i.e. the membership between a patient and a hospital). Such meta-data is private and sensitive by itself; for example, the fact that a sports celebrity visited a hospital before is something that s/he wants to keep confidential since disclosing it may jeopardize his/her future career. A locator service is usually hosted by an untrusted third-party entity, mainly because of the difficulty to find a party unanimously trusted by all the autonomous providers [1]; for example, consider the U.S. government as a candidate, but various scandals including the recent PRISM program [11] have made the government lose the public trust. It is therefore desirable to preserve data privacy

[1]In this paper, we will use "provider" and "hospital" interchangeably.

when the locator service is hosted by an untrusted entity.

PPI techniques (i.e. privacy preserving index [12], [13], [14]) is promising for preserving privacy of a locator service. The PPI design, while originally proposed in domains other than information networks, could protect patient's privacy regarding the medical-history meta-data. When applying PPI for the locator service, the working (also elaborated in Section II) is a two-phase search procedure. Illustrated in Figure 1, a searcher for records of a certain owner [2] first queries the PPI, and obtains a list of providers that may or *may not* have the records of interest. Then for each provider in the list, the searcher attempts to get authenticated and authorized before she can locally search the private records there. In a PPI, the privacy preservation comes from the fact that a searcher may encounter in a PPI result some noise providers (from which she does not find any matching data).

Quantitatively Personalized Privacy Preservation

While existing PPI's have addressed privacy preservation, none of these approaches recognize the needs of personalized privacy, that is, to personalize privacy preservation for different owners and providers. Recall that the privacy of a PPI system is about "an owner t_j has the records stored on provider p_i". It is evident that disclosing the private fact regarding different owners and providers causes different levels of privacy concerns. For example, a woman may consider her visit to a women's health center (e.g., for an abortion) much more sensitive than her visit to a general hospital (e.g., for cough treatment). Similarly, different owners may have different levels of concerns regarding their privacy: While an average person may not care too much about his/her visit to a hospital, a celebrity may be much more concerned about it, because even a small private matter of a celebrity can be publicized by the media (e.g., by paparazzi). It is therefore critical to personalize privacy protection in a PPI system. That being said, using existing PPI approaches can not provide quantitative guarantees on the privacy preservation degree, let alone on a personalized basis. The cause, largely due to the privacy-quality-agnostic way of constructing PPI systems, is analyzed in Appendix B.

In this paper, we propose a new PPI abstraction for quantitatively personalized privacy control, coined ϵ-PPI. Here, ϵ is a privacy-aware knob that allows each owner to mark a personalized privacy level for a data unit delegated to the providers. Specifically, ϵ_j is a value in a spectrum from 0 to 1, where value 0 is for the least privacy concern (in this case, the PPI returns the list of the "true positive" providers who truly have the records of interest) and value 1 for the best privacy preservation (in this case, PPI returns all providers, and a search is essentially broadcast to the whole network). By this means, an attacker observing the PPI search result can only have a bounded confidence by ϵ in successfully identifying a true positive (and vulnerable) provider from the obscured provider list.

To construct the new ϵ-PPI from a network of mutually untrusted providers, we rely on MPC technique (i.e., secure multi-party computation [15], [16], [17], [18]) which addresses the input-data privacy in a generic computation process. However, by directly applying MPC to our ϵ-PPI-construction problem, it raises performance issues. On the one hand, current MPC platforms can only scale to small workloads [19]; they are practical only for simple computation among few parties. On the other hand, a typical ϵ-PPI construction may involve millions of owners and thousands of providers (e.g. in United States there are about six thousand hospitals), which entails an intensive use of bit-wise MPC. It is therefore critical to devise a practical MPC protocol to efficiently carry out the ϵ-PPI construction. In this regards, we propose to minimize the expensive MPC by using a parallel secure sum protocol. The secure sum can be efficiently carried out by a proposed secret sharing scheme with additive homomorphism. Based on the proposed MPC primitive, our index construction protocol protects providers' privacy and can tolerate collusion of up to c providers (c is configurable).

The contributions of this paper are following:

- We propose ϵ-PPI that personalizes the privacy protection with quantitative guarantees. The ϵ-PPI exposes a new delegate operation to owners, which allows them to specify their different levels of privacy concerns. This new privacy knob, coined ϵ, can give quantitative privacy control while enabling information sharing.
- We propose ϵ-PPI construction protocol for an untrusted environment. As far as we know, this is the first PPI construction protocol without assumption on trusted parties or mutual trust relationships between providers. The performance of ϵ-PPI construction protocol is extensively optimized by reducing the use of costly generic MPC and using the proposed domain-specific protocols. The proposed construction protocol is implemented and evaluated with verified performance superiority.
- We introduce a new privacy attack (called common-identity attack) that can break generic PPI systems. The new attack model targets vulnerable common owners/patients who visited a large number of hospitals. Our proposed ϵ-PPI is the first to resist common-identity attacks by using a proposed term-mixing protocol.

The rest of this paper proceeds as follows: Section II formulates the ϵ-PPI problem. Section III and IV respectively describe the computation model and distributed implementation of the ϵ-PPI construction protocol. Section V presents evaluation results, and Section VI surveys the related work before the conclusion in Section VII.

II. Problem Formulation

A. System Model

We formally describe our system model, which involves four entities: 1) a set of n data owners, each of whom, identified by t_j, holds a set of personal records, 2) an information network consisting of m providers in which a provider p_i is an autonomously operating entity (e.g. a hospital), 3) a global PPI server in a third-party domain, 4) a data searcher who wants to find all the records of an owner of interest. The interactions

[2]In this paper we will use "owner" and "patient" interchangeably.

between these four entities are formulated by the following four operations.

- Delegate($\langle t_j, \epsilon_j \rangle$, p_i): A data owner t_j can delegate his/her records to provider p_i based on the trust relationship (e.g. such trust can be built by the previous visit to a hospital). Along with the record delegation, the owner can specify her personal preference in privacy by degree ϵ_j. Here ϵ_j indicates the level of privacy concerns, ranging from 0 up to 1. For example, a VIP user (e.g. a celebrity patient in the eHealthcare network) may want to set the privacy level at a high value while an average patient may set the privacy level at a medium value [3].
- ConstructPPI($\{\epsilon_j\}$): After data records are populated, all m providers in the network join a procedure ConstructPPI to collectively construct the privacy preserving index. The index construction should comply with owner-specified privacy degree $\{\epsilon_j\}$. As will be elaborated, the constructed PPI contains noises or false positives for the purpose of privacy preservation and $\{\epsilon_j\}$ is materialized as the false positive rate of owner t_j.
- QueryPPI(t_j)$\rightarrow$ $\{p_i\}$: At the service time, a searcher s, in the hope of finding owner t_j's records, initiates a two-phase search procedure consisting of two operations, QueryPPI(t_j)$\rightarrow$ $\{p_i\}$ and AuthSearch($s, \{p_i\}, t_j$). This is illustrated in Figure 1. The first phase involves with the locator service in which the searcher poses query request, QueryPPI(t_j), and the PPI server returns a list of providers $\{p_i\}$ who may or may not have records of the requested owner t_j. The query evaluation in PPI server is trivial since the PPI, once constructed, contains the (obscured) mapping between providers and owners.
- AuthSearch(s, $\{p_i\}$, t_j): The second phase in the search is for searcher s to contact each provider in list $\{p_i\}$ (i.e. the result list from the first phase) and to find owner t_j's records there. This process involves user authentication and authorization regarding searcher s; we assume each provider has already set up its local access control subsystem for authorized access to the private personal records. Only after authorization can the searcher search the local repository on provider p_i.

We describe the internal data model in a PPI. Each personal record contains an owner identity t_j[4] (e.g. the person's name). As shown in Figure 2, a provider p_i summarizes its local record repository by a membership vector $M_i(\cdot)$; it indicates the list of owners who have delegated their records on provider p_i. For example, provider p_0 who has records of owner t_0 and t_1 maintains a membership vector as $M_i = \{t_0 : 1, t_1 : 1, t_2 : 0\}$. In our model, the same owner can have records spread across multiple providers (e.g., a patient can visit multiple hospitals). The constructed PPI maintains a mapping between providers and owners; it is essentially a combination of all provider-wise membership data, yet with noises. The PPI mapping data is an $m \times n$ matrix $M'(\cdot, \cdot)$, in which each row is of an owner, each column of a provider and each cell of a Boolean value to indicate the membership/non-membership of the owner to the provider. For the purpose of privacy preservation, there are noises or false positives added in the matrix; for example, regarding provider p_1 and owner t_0, value 1 in the published PPI M is a false positive in the sense that provider p_1 does not have any records of owner t_0 but falsely claims to do so. The false positive value is helpful for obscuring the true and private membership information.

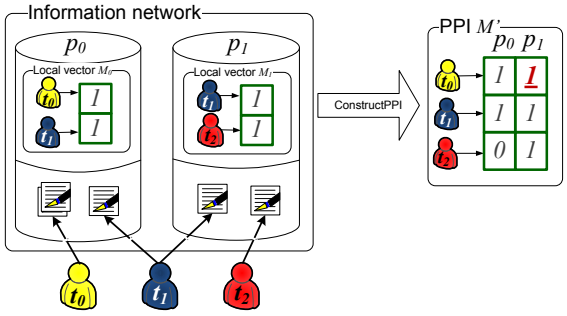


Fig. 2: ϵ-PPI model

Table I summarizes the notations that will be used throughout the rest of the paper.

TABLE I: Notations

Symbols of system model			
t_j	The j-th owner (identity)	n	Number of owners
ϵ_j	Privacy degree of t_j		
p_i	The i-th provider	m	Number of providers
$M_i(\cdot)$	Local vector of p_i	$M'(\cdot, \cdot)$	Data matrix in the PPI
Symbols of ϵ-PPI construction			
β_j	Publishing probability of t_j	σ_j	Frequency of owner t_j
λ	Percentage of common owners	fp_j	Achieved false positive rate of t_j

B. Threat Model

Privacy goals: In our work, we are mainly concerned with the owner-membership privacy; for an owner t_j, the owner-membership privacy is about which providers the owner t_j's records belong to, that is, $M(i, j) = 1$[5]. As mentioned, knowing this information, one can learn about private personal knowledge. Other privacy goals related to the PPI system but not addressed in this work include searcher anonymity and content privacy. The searcher anonymity prevents an attacker from knowing which owner(s) a searcher has searched for, which can be protected by various anonymity protocols [20]. The content privacy [12] involves the detailed content of an owner's record.

In order to attack the owner-membership privacy, we consider a threat model in which an attacker can exploit multiple information sources through different channels. In particular, we consider the following privacy-threatening scenarios:

- **Primary attack** : The primary attack scenario is that an attacker randomly or intentionally chooses a provider p_i and an owner t_j, and then claims that "owner t_j has delegated the records to provider p_i". To determine which providers and owners to attack, the attacker learns about

[3]To prevent every owner from setting the highest value of ϵ, a possible solution is to charge more when the owner sets a higher value of ϵ_j. It is reasonable since in a PPI system higher privacy settings comes with more search overhead.

[4]In this paper, we use "owner" and "identity" interchangeably.

[5]We use $M.(\cdot)$ and $M(\cdot, \cdot)$ interchangeably.

the publicly available PPI data M', and attacks only those with $M'(i,j) = 1$. Given an owner t_j, the attacker can randomly or intentionally (e.g. by her prior knowledge) picks a provider p_i so that $M'(i,j) = 1$. To further refine the attack and improve the confidence, the attacker can exploit other knowledge through various channels, such as colluding providers. Due to space limit, we focus on the attack through the public channel in this paper (the colluding attack and analysis can be found in the tech report [21]).

- **Common-identity attack** : This attack focuses on the common identity which appears in almost all providers in the network. The attacker can learn about the truthful frequency of owner identity σ_j from the public PPI matrix M' (as will be analyzed many PPI's [22], [12], [13] reveals the truthful frequency) and choose the owners with high frequency. By this means, the attacker can have better confidence in succeeding an attack. For example, consider the following extreme case: By learning an owner identity is with frequency $\sigma_j = 100\%$, the attacker can choose any provider and be sure that the chosen provider must be a true positive (i.e., $M(i,j) = 1$).

This paper focuses on attacks on a single owner, while a multi-owner attack boils down to multiple single-owner attacks.

C. Privacy Metric and Degrees

Privacy metric: We measure the privacy disclosure by the attacker's confidence that the attack can succeed. Formally, given an attack on an owner t_j and provider p_i, we measure the privacy disclosure by the probability that the attack can succeed, that is, $Pr(M(i,j) = 1 | M'(i,j) = 1)$. To measure the privacy protection level of a specific owner t_j, we use the average probability of successful attacks against all possible providers that are subject to $M'(i,j) = 1$. The privacy metric is formulated as following.

$$\begin{aligned} Pr(M(\cdot,j)|M'(\cdot,j)) &= \underset{\forall i, M'(i,j)=1}{\text{AVG}} \big(Pr(M(i,j) = 1|M'(i,j) = 1)\big) \\ &= 1 - fp_j \end{aligned}$$

Here, fp_j is the false positive rate of providers in the list of providers $M'(i,j) = 1$. The privacy disclosure metric on owner t_j is equal to $1 - fp_j$, because the false positive providers determines the probability that an attack can succeed/fail. For example, if the list $\{p_i | M(i,j) = 1\}$ is completely without any false positive providers (i.e. $fp_j = 0\%$), then attacks on any provider can succeed, leading to $100\% = 1 - fp_j$ success probability/confidence.

Based on the privacy metric, we further define four discrete privacy degrees. The definition of privacy degrees are based on an information flow model of our privacy threat model, in which an attacker obtains information from the information source through different channels.

- **UNLEAKED**: The information can not flow from the source (i.e. the original record delegated to a provider), and the attacker can not know the information. This is the highest privacy protection level.
- **ϵ-PRIVATE**: The information can flow to attackers through the channel of public PPI data or PPI construction process. If this occurs, the PPI design protects privacy from being disclosed. The PPI can provide a quantitative guarantee on the privacy leakage. Formally, given a privacy degree ϵ_j, it requires the quantitative guarantee as follows.

$$Pr(M(\cdot,j)|M'(\cdot,j)) \leq 1 - \epsilon_j \quad (1)$$

  In particular, when $\epsilon = 0\%$, the attacker might be 100% confident about a successful attack, and privacy is definitely leaked.
- **NOGUARANTEE**: The information can flow to the attacker and the PPI design can not provide any guarantee on privacy leakage. That is, the achieved value of privacy leakage metric may be unpredictable.
- **NOPROTECT**: The information can flow to the attacker and the PPI design does not address the privacy preservation. That is, the privacy is definitely leaked and the attack can succeed with 100% certainty. This is equivalent to the special case of NOGUARANTEE where $\epsilon_j = 0\%$. This is the lowest level of privacy preservation.

Based on our privacy model and metric, we can summarize the prior work in Table II. Due to the space limitation, we put the analysis in Appendix B.

TABLE II: Comparison of ϵ-PPI against existing PPI's

	Primary attack	Common-identity attack
PPI [12], [13]	NOGUARANTEE	NOGUARANTEE
SS-PPI [22]	NOGUARANTEE	NOPROTECT
ϵ-PPI	ϵ-PRIVATE	ϵ-PRIVATE

D. Index Construction of Quantitative Privacy Preservation

In the ϵ-PPI, we aim at achieving ϵ-PRIVATE on a per-identity basis (i.e. personalizing privacy preservation for different owners). The formal problem that this paper address is the index construction with quantitative privacy preservation, which is stated as below.

Proposition 2.1: Consider a network with m providers and n owners; each provider p_i has a local Boolean vector M_i of its membership of n owners. Each owner t_j has a preferred level of privacy preservation ϵ_j. The problem of quantitatively personalized privacy preservation is to construct a PPI that can bound any attacker's confidence (measured by our per-owner privacy metric) under ϵ_j, with regards to all attacks on owner t_j as described in our threat model.

III. ϵ-PPI Construction: Computation

Our ϵ-PPI construction is based on a proposed two-phase framework in which providers first collectively calculate a global value β, and then each provider independently publishes its local vector randomly based on probability β. This framework requires complex computations. In this section, we introduce them at different granularity: We first overview our two-phase construction framework and then introduce the first phase (called the β calculation) in details. At last, we conduct the privacy analysis.

A. A Two-Phrase Construction Framework

We propose a two-phase framework for the ϵ-PPI construction. First, for each owner identity t_j, all m providers collectively calculate a probability value β_j. In the second phase, the private membership value regarding owner t_j and every provider p_i is published. In this paragraph, we assume β_j is already calculated and we focus on describing the second phase – how to use β_j to publish private data. Recall that in our data model, each provider p_i has a Boolean value $M(i,j)$ that indicates the membership of owner t_j in this provider. After knowing value of β_j, provider p_i starts to publish this private Boolean value by randomly flipping it at probability β_j. To be specific, given a membership Boolean value (i.e. $M(i,j)=1$), it is always truthfully published as 1, that is, $M'(i,j)=1$. Given a non-membership value (i.e. $M(i,j)=0$), it is negated to $M'(i,j)=1$ at probability β_j. We call the negated value as the false positive in the published ϵ-PPI. The following formula describes the randomized publication. Note when Boolean value $M(i,j)=1$, it is not allowed to be published as $M'(i,j)=0$.

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

The truthful publication rule (i.e. $1 \rightarrow 1$) guarantees that relevant providers are always in the QueryPPI result and the 100% query recall is ensured. The false-positive publication rule (i.e. $0 \rightarrow 1$) adds noises or false positives to the published PPI which can help obscure the true owner-to-provider membership and thus preserves owner-membership privacy. For multiple owners, different β's are calculated and the randomized publication runs independently.

An example: Consider the case in Figure 2. For owner t_0, if β_0 is calculated to be 0.5, then provider p_1 would publish its negative membership value $M_1(0)=0$ as value 1 with probability 0.5. In this example, it is flipped and the constructed ϵ-PPI contains $M'(1,0)=1$. Similarly for identity t_2 and provider p_0, it is also subject to flipping at probability β_2. In this example, it is not flipped, and the constructed ϵ-PPI contains $M'(0,2)=0$.

B. The β Calculation

In the randomized publication, β_j determines the amount of false positives in the published ϵ-PPI. For quantitative privacy preservation, it is essential to calculate a β_j value that makes the false positive amount meet the privacy requirement regarding ϵ_j. In this part, we focus on describing the calculation of β which serves as the first phase in ϵ-PPI construction process. Concretely we consider two cases: the common identity case and the non-common identity case. Recall that the common identity refers to such an owner who delegates her records to almost all providers in the network. The general PPI construction is vulnerable to the common-identity attack and it needs to be specially treated.

1) The Case of Non-common Identity

In the case of non-common identity, negative providers suffice to meet the desired privacy degree. We consider the problem of setting value β_j for identity t_j in order to meet the desired ϵ_j. Recall the randomized publication: Multiple providers independently runs an identical random process, and this can be modeled as a series of Bernoulli trials (note that the publishing probability $\beta(t_j)$ is the same to all providers). Our goal is to achieve privacy requirement that $fp_j \geq \epsilon_j$ with high level success ratio p_p, that is, $p_p = Pr(fp_j \geq \epsilon_j)$. Under this model, we propose three policies to calculate β with different quantitative guarantees: a basic policy β_b that guarantees $fp_j \geq \epsilon_j$ with 50% probability, and an incremented expectation based policy β_d, and a Chernoff bound based policy β_c that guarantees $fp_j \geq \epsilon_j$ with γ probability where success ratio γ can be configured.

Basic policy: The basic policy sets the β value so that the expected amount of false positives among m providers is satisfactory, that is, be at least $\epsilon_j \cdot m(1-\sigma_j)$. Formally,

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

The basic policy has poor quality in attaining the desired privacy preservation; the actual value fp_j is bigger than ϵ_j with only 50% success ratio.

Incremented expectation-based policy: The incremented expectation-based approach is to increase the expectation-based $\beta_b(t_j)$ by a constant value, that is,

$$\beta_d(t_j) = \beta_b(t_j) + \Delta \tag{4}$$

Incremental Δ can be configurable based on the quality requirement; the bigger the value is, the higher success ratio p_p is expected to attain. However, there is no direct connection between the configured value of Δ and the success ratio p_p that can be achieved, leaving it a hard task to figure out the right value of Δ based on desired p_p.

Chernoff bound-based policy: Toward an effective policy to calculate β, we apply the Chernoff bounds to the randomized publication process which is modeled as Bernoulli trials. This policy allows direct control of the success ratio. Formally, it has the property described in Theorem 3.1 (with the proof in Appendix A-A).

Theorem 3.1: Given desired success ratio $\gamma > 50\%$, let $G_j = \frac{\ln \frac{1}{1-\gamma}}{(1-\sigma_j)m}$ and

$$\beta_c(t_j) \geq \beta_b(t_j) + G_j + \sqrt{G_j^2 + 2\beta_b(t_j)G_j} \tag{5}$$

Then, randomized publication with $\beta(t_j)=\beta_c(t_j)$ statistically guarantees that the published ϵ-PPI can meet privacy requirement $fp_j \geq \epsilon_j$ with success ratio larger than γ.

2) The Case of Common Identities

With the above β calculation for non-common identities, the constructed ϵ-PPI is vulnerable to the common-identity attack. Because the β_*[6] bears information of identity frequency σ_j, and during our index construction framework, β needs to be released to all participating providers. A colluding provider would release such information to the attacker who can easily obtain the truthful identity frequency σ (e.g., from Equation 3 assuming ϵ_j is publicly known) and effectively formulates the common-identity attack.

To defend against the common-identity attack, ϵ-PPI construction employs an identity-mixing technique for common identities. The idea is to mix common identities with certain non-common identities by exaggerating the calculated β_j (i.e. falsely increasing certain β_j to 100%) from which one can not distinguish common identities from the rest. To be specific, for a non-common identity t_j, we allow its β_j to be exaggerated to 100% with probability λ, that is,

$$\beta = \begin{cases} \begin{cases} \beta_*, & 1-\lambda \\ 1, & \lambda \end{cases}, & \beta_* < 1 \\ 1, & \beta_* \geq 1 \end{cases} \quad (6)$$

Given a set of common identities, we need to determine how many non-common identities should be chosen for mixing, in other words, to determine the value of λ. While a big value of λ can hide common identities among the non-common ones, it incurs unnecessarily high search cost. On the other hand, a value of λ which is too small would leave common identities unprotected and vulnerable. In ϵ-PPI, we use the following heuristic-based policy to calculate λ.

- In the set of mixed identities, the percentage of non-common identities should be no smaller than ξ. Since there are $\sum_{\beta_* \geq 1} 1$ common identities and thus $\sum_{\beta_* < 1} \lambda$ non-common identities in the set, we have the following formula.

$$\begin{aligned} \xi &\leq \frac{\sum_{\beta_*<1} \lambda}{\sum_{\beta_* \geq 1} 1 + \sum_{\beta_*<1} \lambda} \quad (7) \\ \Rightarrow \lambda &\geq \frac{\xi}{1-\xi} \cdot \frac{\sum_{\beta_* \geq 1} 1}{n - \sum_{\beta_* \geq 1} 1} \end{aligned}$$

3) β Calculation: Putting It Together

We summarize the β calculation in the ϵ-PPI construction. For each identity t_j, $\beta(t_j)$ is calculated based on Equation 6, which follows the computation flows as below. The underline symbol indicates the variable is private and $\Rightarrow$ indicates the computation is fairly complex and heavy (e.g. involving square root when calculating β_*).

$$\begin{aligned} &\text{Frequency } \underline{\sigma} \Rightarrow \text{Raw probability } \underline{\beta_*} \rightarrow \quad (8) \\ &\rightarrow \textstyle\sum_{\underline{\beta_* \geq 1}} 1 \rightarrow \text{Common id percentage } \lambda \rightarrow \text{Final probability } \beta \end{aligned}$$

[6]We use β_* to denote the probability value calculated by any of the three policies for non-common identities.

C. Privacy Analysis of Constructed ϵ-PPI

We present the privacy analysis of the constructed ϵ-PPI under our threat model.

Privacy under primary attack: The property of the three policies of calculating β_* suggests that the false positive rate in the published ϵ-PPI should be no smaller than ϵ_j in a statistical sense. Recall that the false positive rate bounds the attacker's confidence; it implies that ϵ-PPI achieves an ϵ-PRIVATE degree against the primary attack. It is noteworthy that our ϵ-PPI is fully resistant to repeated attacks against the same identity over time, because the ϵ-PPI is static; once constructed and having privacy protected, it stays the same.

Privacy under common-identity attack: For common-identity attack, the attacker's confidence in choosing a true common identity depends on the percentage of true common identities among the (mixed) common identities in the published ϵ-PPI. Therefore the privacy preservation degree is bounded by the percentage of false positives (in this case, it depends on the percentage of the non-common identities which is mixed and published as common identities in the published ϵ-PPI), which equals ξ. By properly setting λ, we can have $\xi = \max_{\forall t_j \in \{\text{common identities}\}} \epsilon_j$. By this way, it is guaranteed to achieve the per-identity ϵ-PRIVATE degree against the common-identity attack.

IV. ϵ-PPI Construction: Realization

This section describes the design and implementation of a distributed and secure protocol that realizes the computation of ϵ-PPI construction described in the previous section.

A. Challenge and Design

The goal of our protocol is to efficiently and securely compute the publishing probability $\{\beta_j\}$ among a set of mutually untrusted providers who are reluctant to exchange the private membership vector with others. The computation is based secure multi-party computation (or MPC) that protects the input-data privacy. It is challenging to construct ϵ-PPIusing MPC in a large information network. On the one hand, current techniques for MPC only support small computation workloads [19]. On the other hand, the computation required in ϵ-PPI construction is big and complex; the computation model involves large number of identities and providers; even for a single identity it involves fairly complex computation (e.g., square root and logarithm as in Equation 5). This poses a huge challenge to design a practical protocol for secure ϵ-PPI construction.

To address the above challenge, we propose an efficient and secure construction protocol based on the principle of *minimizing the secure computation*. Given a computation flow in Equation 8, our protocol design has three salient features: 1) It separates the secure and non-secure computations by the last appearance of private variables in the flow (note that the computation flows from the private data input to the end of non-private result). 2) It reorders the computation to minimize the expensive secure computation. The idea is to push down complex computation towards the non-private end. To be specific, instead of first carrying out complex floating

point computations for raw probability β, as in Formula 8, we push such computations down through the flow and pull up the obscuring computations for private input, as in Formula 9. 3) To scale to a large number of providers, we propose an efficient protocol for calculating the secure sum, and use it to reduce the "core" of the MPC part.

$$\underline{\sigma} \rightarrow \quad \sum_{\underline{\sigma}<\sigma'} 1 \quad \rightarrow \lambda \rightarrow \begin{cases} \rightarrow & \beta = 1 \\ \Rightarrow & \beta = \beta_* \end{cases} \tag{9}$$

B. The Distributed Algorithm

Following our design, we propose a practical distributed algorithm to run the two-phase ϵ-PPI construction. The overall workflow is illustrated in Figure 3. For simplicity, we focus on phase 1 for β calculation. The β calculation is realized in two stages by itself: As illustrated in Algorithm 1, the first stage is a SecSumShare protocol which, given m input Boolean from the providers, outputs c secret shares whose sum is equal to the sum of these m Boolean. Here, c is the number of shares that can be configurable based on the tolerance on provider collusion. The output c shares have the security property that a party knowing $x < c$ shares can not deduce any information about the sensitive sum of m Boolean. For different identities, the SecSumShare protocol runs multiple instances independently and in parallel, which collectively produce c vectors of shares, denoted by $s(i, \cdot)$, where $i \in [0, c-1]$. The c vectors are distributed to c coordinate providers (for simplicity we assume they are providers $p_0, \ldots, p_{c-1}$) on which the second-stage protocol, CountBelow, is run. As shown by Algorithm 2, given c vectors $s(0, \cdot), \ldots s(c-1, \cdot)$ and a threshold t, the CountBelow algorithm sums them to vector $\sum_i s(i, \cdot)$ and counts the number of elements that are bigger than t.

TABLE III: Distributed algorithms for ϵ-PPI construction

Algorithm 1 calculate-beta($M_0, \ldots M_{n-1}$)

1: $\{s(0, \cdot), \ldots s(c-1, \cdot)\} \leftarrow$ SecSumShare($M_0, \ldots M_{n-1}$)
2: $\sigma'(\cdot)$ is calculated under condition $\beta_* = 1$, by either Equation 3, or 4 or 5.
3: $\sum_{\sigma \geq \sigma'} 1 \leftarrow$CountBelow$(s(0, \cdot), \ldots s(c-1, \cdot), \sigma'(\cdot) \cdot m)$
4: $\{\beta_0, \ldots \beta_{m-1}\} \leftarrow \sum_{\sigma \geq \sigma'} 1$ ▷ By Equation 9

Algorithm 2 CountBelow($s(0, \cdot), \ldots s(c-1, \cdot)$, threshold t)

1: count$\leftarrow 0$
2: **for** $\forall j \in [0, m-1)$ **do**
3: $\quad S[j] \leftarrow \sum_i s(i, j)$
4: $\quad$ **if** $S[j] < t$ **then**
5: $\quad\quad$ count++
6: $\quad$ **end if**
7: **end for**
8: **return** count

1) Distributed Algorithm for SecSumShare

We use an example in the top box in Figure 3 to illustrate the distributed algorithm of SecSumShare. In the example $c = 3$ and there are five providers $p_0, \ldots p_4$. The example focuses on a single identity case for t_j (e.g. $j = 0$). Out of the 5 providers, p_1 and p_2 have records of owner t_0 (i.e., $M(1,0) = M(2,0) = 1$). SecSumShare requires modular operations;

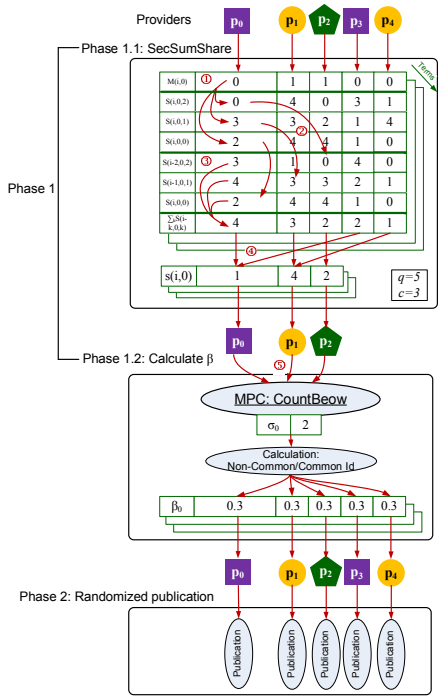


Fig. 3: An example of ϵ-PPI construction algorithm

in this example, the modulus divisor is $q = 5$. It runs in the following 4 steps.

1 Generating shares: each provider p_i decomposes its private input Boolean $M(i, j)$ into c shares, denoted by $\{S(i, j, k)\}$, with $k \in [0, c-1]$. The first $c-1$ shares are randomly picked from interval $[0, q]$ and the last share is deterministically chosen so that the sum of all shares equals the input Boolean $M(i, 0)$ in modulo q. That is, $(\sum_{k \in [0,c]} S(i, j, k)) \mod q = M(i, j)$. In Figure 3, as depicted by arrows ①, p_0's input $M(0, 0)$ is decomposed to $c = 3$ shares, $\{S(0, 0, k)|k\} = \{2, 3, 0\}$. It ensures $(2 + 3 + 0) \mod 5 = 0$.

2 Distributing shares: each provider p_i then distributes its shares to the next $c-1$ neighbor providers; k-th shares $S(i, j, k)$ will be sent out to k-th successor of provider p_i, that is, $p_{(i+k) \mod m}$. As shown by arrows ② in Figure 3, p_0 keeps the first share 2 locally, sends its second share 3 to its successor p_1 and the third share 0 to its 2-hop successor p_2.

3 Summing shares: each provider then sums up all shares she has received in the previous step to obtain the *super-share*. In Figure 3, after the step of share distribution, provider p_0 receives 3 from p_3, 4 from p_4 and 2 from herself. As depicted by arrows ③, the super-share is calculated to be $3 + 4 + 2 \mod 5 = 4$.

4 Aggregating super-shares: each provider sends its super-share to a set of c coordinators. These coordinators receiving super-shares then sum the received shares up and output the summed vector $s(i, \cdot)$ to the next-stage CountBelow protocol. In Figure 3, provider p_0, p_1, p_2 are chosen as coordinators and arrow ④ shows that the sum of super-shares on provider p_0 is $s(0, 0) = (4 + 2)$

$\mod 5 = 1$. The sum of all the values on coordinators should be equal to the number of total appearances of identity t_0. That is, $1+4+2 \mod 5 = 2$. Note that there are two providers with identity t_0. This total appearance number or identity frequency may be sensitive (in the case of common identity) and can not be disclosed immediately, which is why we need the second stage protocol, CountBelow.

2) Implementation of CountBelow computation

The secure computation of CountBelow (in Algorithm 2) is implemented by using a generic MPC protocol. Each party corresponds to a coordinate provider in the ϵ-PPI system. Specifically, we choose a Boolean-circuit based MPC protocol FairplayMP [16] for implementation. Since Algorithm 2 is implemented by expensive MPC it normally becomes the bottleneck of the system; in practice, $c \ll m$ and thus the network can scale to large number of providers m while the MPC is still limited to a subset of the network.

C. Privacy Analysis of Constructing ϵ-PPI

We analyze the privacy preservation of ϵ-PPI construction process. We mainly consider a semi-honest model, which is consistent with the existing MPC work [16]. The privacy analysis is conducted from three aspects: 1) The privacy guarantee of SecSumShare protocol. It guarantees: 1.1) $(2c-3)$-secrecy of input privacy [22]: With less than c providers in collusion, none of any private input can be learned by providers other than its owner. 1.2) c-secrecy of output privacy: The private sum can only be reconstructed when all c shares are used. With less than c shares, one can learn nothing regarding the private sum. The output privacy is formally presented in Theorem 4.1 with proof in Appendix A-B. 2) The security and privacy of CountBelow relies on that of the MPC used in implementation. The generic MPC technique can provide information confidentiality against c colluding providers [16]. 3) The final output β does not carry any private information, and is safe to be released to the (potentially untrusted) providers for the randomized publication.

Theorem 4.1: The SecSumShare's output is a (c, c) secret sharing scheme. Specifically, for an owner t_j, SecSumShare protocol outputs c shares, $\{s(i,j)|\forall i \in [0, c-1]\}$, whose sum is the secret v_j. The c shares have the following properties.

- *Recoverability*: Given c output shares, the secret value v_j (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, in the sense that the value's conditional distribution given the known shares is the same as the prior distribution,

$$\forall x \in \mathbb{Z}_q, Pr(v_j = x) = Pr(v_j = x | V \subset \{s(i,j)\}))$$

where V is any proper subset of $\{s(i,j)\}$.

V. Experiments

To evaluate the proposed ϵ-PPI, we have done two set of experiments: The first set, based on simulations, evaluates how effective the ϵ-PPI can be in terms of delivering quantitative privacy protection. The second set evaluates the performance of our index construction protocol. For realistic performance results, we have implemented a functioning prototype for ϵ-PPI construction.

A. Effectiveness of Privacy Preservation

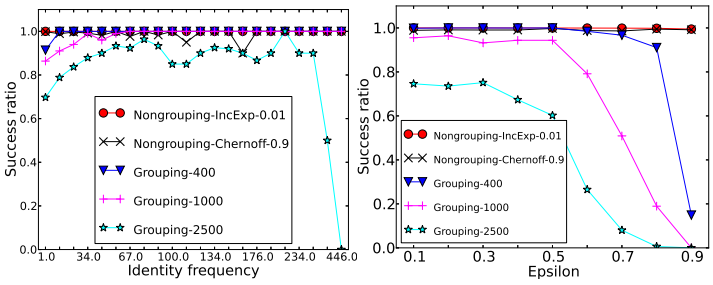
Experimental setup: To simulate the information network, we use a distributed document dataset [23] of $2,500 - 25,000$ small digital libraries, each of which simulates a provider in our problem setting. To be specific, this dataset defines a "collection" table, which maintains the mapping from the documents to collections. The documents are further derived from NIST's publicly available TREC-WT10g dataset [24]. To adapt to our problem setting, each collection is treated as a provider and the source web URLs (as defined in TREC-WT10g dataset) of the documents are treated as owner's identity. If not otherwise specified, we use no more than $10,000$ providers in the experiments. Using the collection table, it also allows us to emulate the membership matrix M. The dataset does not have a privacy metric for the query phrase. In our experiment, we randomly generate the privacy degree ϵ in the domain $[0, 1]$. We use a metric, success ratio, to measure the effectiveness. The success ratio is the percentage of identities whose false positive rates in the constructed PPI are no smaller than the desired rate ϵ_j.

1) ϵ-PPI versus Existing Grouping-based PPI's

The experiments compare ϵ-PPI with existing PPI's. The existing PPI's [12], [13], [22] are based on a grouping technique; providers are organized into disjoint privacy groups so that different providers from the same group are indistinguishable from the searchers. By contrast, ϵ-PPI does not utilize grouping technique and is referred to in this section as a non-grouping approach. In the experiment, we measure the success ratio of privacy preservation, and search performance. Grouping PPI's are tested under different group sizes. Given a network of fixed providers, we use the group number to change average group size. We test grouping PPI with the Chernoff bound-based and the incremented expectation-based policies under the default setting. The expected false positive rate is configured at 0.8, and the number of providers is $10,000$. We uniformly sample 20 times and report the average results.

Results are illustrated in Figure 4. Non-grouping PPI generally performs much better and more stable than the grouping approach in terms of success ratio. With proper configuration (e.g. $\Delta = 0.01$ for incremental expectation-based policy and $\gamma = 0.9$ for Chernoff policy), the non-grouping ϵ-PPI always achieves near-optimal success ratio (i.e. 1.0). By contrast, the grouping PPI's display instability in their success ratio. For example, as shown by the "Grouping (#groups 2000)" series in Figure 4a, the success ratio fluctuates between 0.0 and 1.0, which renders it difficult to provide a guarantee to the system and owners. The reason is that with 2000 groups, sample space in each group is too small (i.e., with 50 providers) to hold a stable result for success ratio. When varying ϵ, similar behavior is shown in Figure 4b; the success ratio of grouping PPI's quickly degrades to 0, leading to unacceptable privacy quality.

This is due to the grouping design of PPI that is agnostic to different owners. This set of experiments shows that the privacy degree of non-grouping PPI's can be effectively tuned, implying the ease of a practical use.



(a) Varying identity frequency (b) Varying ϵ

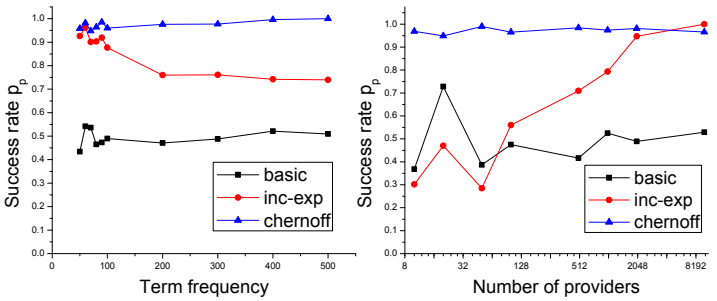
Fig. 4: Comparing non-grouping and grouping

2) Comparing different β-calculation policies

We evaluate and compare the effectiveness of three β-calculation policies with ϵ-PPI. In the experiments, we tested various parameter settings and show the representative results at the following settings: $\Delta = 0.02$ as in the incremented expectation-based policy, expected success ratio $\gamma = 0.9$ as in the Chernoff bound based policy. The default false positive rate is set at $\epsilon = 0.5$. The experiment results measuring the success ratio are reported in Figure 5. In Figure 5a, we vary the identity frequency from near 0 to about 500 providers with totally $10,000$ providers in the network. In Figure 5b we vary the number of providers and set the identity frequency to be constant 0.1. It can be seen from the results that while the Chernoff bound-based policy (with $\gamma = 0.9$) always achieves near-optimal success ratio (i.e., close to 1.0), the other two policies fall short in certain cases; the expectation-based policy is not configurable and achieves the success rate only at 0.5. This is expected because the expectation-based approach works on an average sense. For the incremented expectation-based policy, its success ratio, though approaching 1.0 in some cases, is much smaller than 1.0 and unsatisfactory in other cases (e.g. for terms of high frequency as in Figure 5a and for few providers as in Figure 5b). On the other hand, the high-level privacy preservation of the Chernoff bound policy comes with reasonable search overhead. The related experiment results can be found in technical report [21].

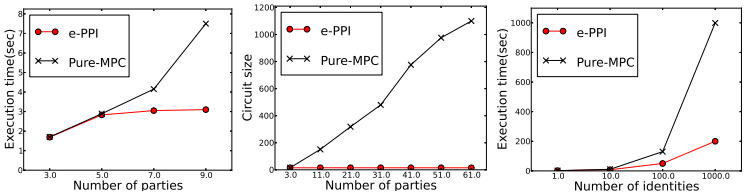
B. Performance of Index Construction

Experimental setup: We evaluate the performance of our distributed ϵ-PPI construction protocol. Towards that, we have implemented a functioning prototype. The CountBelow is implemented by using an MPC software, FairplayMP [16], which is based on Boolean circuits. The implemented CountBelow protocol is written 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 computation. We implement the SecSumShare protocol in Java. In particular, we use a third-party library Netty [25] for network communications



(a) Varying frequency under $10,000$ providers (b) Varying provider numbers under frequency 0.1

Fig. 5: Quality of privacy preservation



(a) Execution time with single identity (b) Circuit size with single identity (c) Scale up identity number

Fig. 6: Performance of index construction protocol

and Google's protocol buffer [26] for object serialization. We conduct experiments on a number of machines in Emulab [27], [28], each equipped with a 2.4 GHz 64-bit Quad Core Xeon processor and 12 GB RAM. In the experiments, the number of machines tested is varied from 3 to 9 (due to limited resource at hand). For each experiment, the protocol is compiled to and run on the same number of parties. Each party is mapped to one dedicated physical machine. The experiment uses a configuration of $c = 3$.

To justify the standpoint of our design that MPC is expensive, we compare our MPC-reduced approach as in the ϵ-PPI construction protocol against a pure MPC approach. The pure MPC approach does not use the SecSumShare protocol to reduce the number of parties in the generic MPC part and directly accepts inputs from the m providers. The metric used in the experiment is the start-to-end execution time, which is the time duration from when the protocol starts to run to when the last machine reports to finish. The result is shown as in Figure 6a. It can be seen that the pure MPC approach generally incurs longer execution time than our MPC-reduced approach (used in ϵ-PPI construction): As the information network grows large, while the execution time of pure MPC approach increases super-linearly, that of MPC-reduced approach increases slowly. This difference is due to the fact that the MPC in our MPC-reduced approach is fixed to c parties and does not change as the number of providers m grows. And the parallel SecSumShare in MPC-reduced approach is scalable in m as well, since each party runs in constant rounds, and each round sends a constant number (at most $c-1$) of messages to its neighbors. For scaling with more parties, we use the metric of circuit size, which is the size of the compiled MPC program. As a valid metric, the circuit size

determines the execution time[7] in real runs. By this means, we can show the scalability result of up to 60 parties as in Figure 6b. Similar performance improvement can be observed except that the circuit size grows linearly with the number of parties involved. Finally, we also study the scalability from running the protocol with multiple identities in a three-party network. The result in Figure 6c shows that ϵ-PPI construction grows with the number of identities at a much slower rate than that of the pure MPC approach.

VI. Related Work

A. Privacy-Preserving Data Indexing

Non-encryption based PPI: PPI is designed to index access controlled contents scattered across multiple content providers. While being stored on an untrusted server, PPI aims at preserving the content privacy of all participant providers. Inspired by the privacy definition of k-anonymity [29], existing PPI work [12], [13], [22] follows the *grouping-based* approach; it organizes providers into disjoint privacy groups, such that providers from the same group are indistinguishable to the searchers. To construct such indexes, many existing approaches [12], [13], [30] assume providers are willing to disclose their private local indexes, an unrealistic assumption when there is a lack of mutual trust between providers. SS-PPI [22] is proposed with resistance against colluding attacks. While most existing grouping PPI's utilize 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's, it lacks per-owner concerns and quantitative privacy guarantees. Moreover, organizing providers in groups usually leads to query broadcasting (e.g, with positive providers scattered in all groups), rendering search performance inefficient. By contrast, ϵ-PPI is a brand new PPI abstraction without grouping (i.e. non-grouping PPI as mentioned before), which provides quantitative privacy control on a per-owner basis.

Index with search-able encryption: Building search-able indexes over encrypted data has been widely studied in the context of both symmetric key cryptography [31] and public key cryptography [32], [33], [34]. In this architecture, content providers build their local indices and encrypt all the data and indices before submitting them to the untrusted server. During query time, the searcher first gets authenticated and authorized by the corresponding content provider; the searcher then contacts the untrusted server and searches against the encrypted index. This system architecture makes the assumption that a searcher already knows which provider possesses the data of her interest, which is unrealistic in the PPI scenario. Besides, unlike the encryption-based system, performance is a motivating factor behind the design of our PPI, by making no use of encryption during the query serving time.

[7]Regarding the detailed definition of circuit size and the exact correlation between circuit size and execution time, it can be found in FairplayMP [16].

B. Secure Distributed Computations

Practical MPC: Recently a large body of research work [15], [16], [17], [18], [35] is dedicated towards a practical MPC platform. Traditional work for generic MPC largely falls under two models, the garbled functions used for Boolean circuits and the homomorphic encryption used for arithmetic calculation. Towards efficient and practical MPC systems, Fairplay [15], [16] implements the computation of Boolean circuits for two or more parties, and VIFF [18] is a runtime for the computation of arithmetic circuits. MightBeEvil [36] supports efficient two-party computation via garbled circuit. Based on the observation that different computation models can lead to performance gain for different workloads, TASTY [17] proposes to use a modular and adaptive design which divides the whole workload into several modules and accordingly maps the modular workload to to a specific MPC model. It is realized by a scheme that can convert the encrypted or garbled data between modules. Recent work [35] extends the domain of practical MPC (from integers) to floating point numbers by using Shamir's secret sharing.

Privacy-preserving multi-source analysis: Based on the primitive provided by MPC, there is a large body of research work on privacy-preserving analysis of multi-source data. We briefly survey this research area. Private record linkage (or PRL) [37], [38] is important in the health information exchanges. The task of PRL is to identify medical records of the same patient which are with semantically heterogeneous demographic information and are distributed across multiple Healthcare providers. In industry, Master Patient Indexes [39], [10] are real-world systems that recently emerge to support PRL. In the research community, privacy-preserving PRL schemes are recently proposed [40], [41]. The PRL technique is complementary to our ϵ-PPI in the sense that they could work together to support a federated search service for patient medical history based on heterogeneously distributed patient data. In the domain of corporate IT business, prior work [42] proposes a privacy-preserving framework for analyzing OLAP workloads for business intelligence (or BI). For data sharing between multiple corporate entities, DJoin [19] is proposed for privacy-preserving join computation. To address the inefficiency of generic MPC, DJoin uses an efficient but domain-specific primitive for set operations [43] in combination with MPC. In similar spirit, our ϵ-PPI construction protocol reduces the expensive MPC by applying the efficient secure-sharing technique.

VII. Conclusion

In this paper, we propose ϵ-PPI for personalized privacy control with quantitative guarantee. ϵ-PPI allows each data owner to specify her personal preference in privacy preservation, and ϵ-PPI can guarantee that such personalized privacy can be preserved in a quantitative fashion. In the design of ϵ-PPI, we identify a vulnerability of generic PPI systems, the common-identity attack. We propose an identity-mixing mechanism to protect ϵ-PPI against such attacks. We have implemented the construction protocol for ϵ-PPI without any

trusted party involved. We optimize the performance of secure index construction protocol by minimizing the use of expensive MPC. We have built a generic privacy threat model and performed security analysis which shows the advantages of ϵ-PPI over other PPI system in terms of privacy preservation quality.

Acknowledgment

The first author thank Mark L. Braunstein and Myung Choi for the discussion on health informatics. This research is partially supported by grants from NSF CISE NetSE program, SaTC program, I/UCRC, an IBM faculty award and a grant from Intel ICST on Cloud Computing.

References

[1] "Nhin: http://www.hhs.gov/healthit/healthnetwork."
[2] "Gahin: http://www.gahin.org/."
[3] "Commonwell: http://www.commonwellalliance.org/."
[4] "Hippa, http://www.cms.hhs.gov/hipaageninfo/."
[5] "Coursera: https://www.coursera.org/."
[6] "Studip, http://www.studip.de/."
[7] "Diaspora: https://joindiaspora.com/."
[8] "Twister: http://twister.net.co/."
[9] "Commonwell rls: http://www.commonwellalliance.org/services."
[10] "Openempi: https://openempi.kenai.com/."
[11] "Prism, http://en.wikipedia.org/wiki/prism_(surveillance_program)."
[12] M. Bawa, R. J. B. Jr., and R. Agrawal, "Privacy-preserving indexing of documents on the network," in *VLDB*, 2003, pp. 922–933.
[13] 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.
[14] 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.
[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] M. Wright, M. Adler, B. N. Levine, and C. Shields, "An analysis of the degradation of anonymous protocols," in *NDSS*, 2002.
[21] Y. Tang, L. Liu, and A. Iyengar, "e-ppi: Searching information networks with quantitative privacy guarantee," *Gerogia Tech Technical Report GIT-CERCS-14-02*.
[22] Y. Tang, T. Wang, and L. Liu, "Privacy preserving indexing for ehealth information networks," in *CIKM*, 2011, pp. 905–914.
[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] "Netty: http://netty.io."
[26] "Protobuf: http://code.google.com/p/protobuf/."
[27] "http://www.emulab.net/."
[28] 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.
[29] 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.
[30] 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.
[31] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE SSP*, 2000, pp. 44–55.
[32] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *ICDCS*, 2010, pp. 253–262.
[33] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in *ICDCS*, 2011, pp. 383–392.
[34] 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.
[35] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele, "Secure computation on floating point numbers," in *NDSS*, 2013.
[36] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *USENIX Security Symposium*, 2011.
[37] S. L. DuVall, A. M. Fraser, K. Rowe, A. Thomas, and G. P. Mineau, "Evaluation of record linkage between a large healthcare provider and the utah population database," *JAMIA*, vol. 19, no. e1, 2012.
[38] T. B. Newman and A. N. Brown, "Use of commercial record linkage software and vital statistics to identify patient deaths," *Journal of the American Medical Informatics Association*, vol. 4, no. 3, pp. 233–237, 1997. [Online]. Available: http://jamia.bmj.com/content/4/3/233.abstract
[39] "Nextgate: http://www.nextgate.com/our-products/empi/."
[40] M. Kuzu, M. Kantarcioglu, E. A. Durham, C. Toth, and B. Malin, "A practical approach to achieve private medical record linkage in light of public resources," *JAMIA*, vol. 20, no. 2, pp. 285–292, 2013.
[41] M. Kuzu, M. Kantarcioglu, A. Inan, E. Bertino, E. Durham, and B. Malin, "Efficient privacy-aware record integration," in *EDBT*, 2013, pp. 167–178.
[42] A. Cuzzocrea and E. Bertino, "A secure multiparty computation privacy preserving olap framework over distributed xml data," in *Proceedings of the 2010 ACM Symposium on Applied Computing*, ser. SAC '10. New York, NY, USA: ACM, 2010, pp. 1666–1673. [Online]. Available: http://doi.acm.org/10.1145/1774088.1774447
[43] L. Kissner and D. Song, "Privacy-preserving set operations," in *in Advances in Cryptology - CRYPTO 2005, LNCS*. Springer, 2005, pp. 241–257.
[44] M. Mitzenmacher and E. Upfal, *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

Appendix A
Proof of theorems

A. Proof of Theorem 3.1

Proof: We model the problem as Bernoulli trials and prove the theorem by applying Chernoff bound. For a term t_j, the total number of false positive providers is modeled as sum of $T = m(1-\sigma_j)$ Bernoulli trials, because there are $m(1-\sigma_j)$ negative providers for term t_j and each negative provider independently and randomly publishes its own bit, a process that can be modeled as a single Bernoulli trials. In the trial, when the negative provider becomes a false positive (i.e., $0 \rightarrow 1$) which occurs at probability $\beta(t_j)$, the Bernoulli random variable, denoted by X, takes on value 1. Otherwise, it takes the value 0. Let $E(X)$ be the expectation of variable X, which in our case is,

$$E(X) = m(1-\sigma_j) \cdot \beta(t_j) \tag{10}$$

We can apply the Chernoff bound for the sum of Bernoulli trials, $Pr(X \leq (1-\delta)E(X)) \leq e^{-\delta^2 E(X)/2}$ [44], where $\delta > 0$ is any positive number. For term t_j, the expected success rate, denoted by $p_p(t_j)$, is equal to the probability of a publication success, that is, $p_p(t_j) = Pr(fp_j > \epsilon_j)$. Note $fp_j = \frac{X}{X+\sigma_j \cdot m}$,

we have,

$$\begin{aligned} p_p(t_j) &= 1 - Pr(fp_j \leq \epsilon_j) \\ &= 1 - Pr(X \leq m\frac{\sigma_j}{\epsilon_j^{-1}-1}) \\ &\geq 1 - e^{-\delta_j^2 m(1-\sigma_j)\beta(t_j)/2} \end{aligned} \quad (11)$$

In here, $\delta_j = 1 - \frac{1}{(\epsilon_j^{-1}-1)(\sigma_j^{-1}-1)} \cdot \frac{1}{\beta(t_j)} = 1 - \frac{\beta_b(t_j)}{\beta(t_j)}$. Recall that γ is the required minimal success rate. If we can have

$$1 - e^{-\delta_j^2 m(1-\sigma_j)\beta(t_j)/2} \geq \gamma \quad (12)$$

for all indexed terms, then $\forall j, p_p(t_j) \geq \gamma$. This means in the case of large number of terms, the percentage of successfully published terms or p_p is expected to be larger than or equal to γ, i.e., $p_p \geq \gamma$, which is the proposition. Hence, by plugging δ_j in Equation 12, we can derive,

$$(\beta_c(t_j))^2 - 2\Big(\beta_b(t_j) + \frac{\ln\frac{1}{1-\gamma}}{(1-\sigma_j)m}\Big)\beta_c(t_j) + (\beta_b(t_j))^2 \geq 0$$

Note $\frac{\ln\frac{1}{1-\gamma}}{(1-\sigma_j)m} = G_j$, and $\beta_c(t_j)$ should be bigger than $\beta_b(t_j)$ since success ratio is larger than 50%. Solving the inequality and taking only the solution that satisfies $\beta_c(t_j) > \beta_b(t_j)$, we have,

$$\beta_c(t_j) \geq \beta_b(t_j) + G_j + \sqrt{G_j^2 + 2\beta_b(t_j)G_j}$$

■

B. Proof of Theorem 4.1

Proof: Recoverability can be trivially proved based on the fact that $\sum_{\forall i \in [0,c-1]} s(i,j) = v_j$.

To prove secrecy, we examine the process of generating super-shares $s(i,j)$. It is easy to see that the SecSumShare protocol uses a (c,c) secret sharing to split each private input $M(i,j)$. The generated c shares for each input value are distributed to c different output super-shares. For each private input $M(i,j)$, an output super share $s(i,j)$ has included *one and only one* share from it. Therefore, when an adversary knows at most $c-1$ outputs, at least one share of each private input is still unknown to her. This leaves the value of any input completely undetermined to this adversary, thus the secret or the sum of input values completely undetermined. ■

Appendix B
Analysis of Conventional PPIs

We analyze the privacy of existing PPI work and compare it with that of ϵ-PPI. Here, we consider the primary attack and the common-term attack . Before that, we briefly introduce the construction protocol of existing PPI. To be consistent with terminology, we use term to refer to owner's identity in this section, for example, the common-identity attack is referred to as the common-term attack.

Grouping PPI: Inspired by k-anonymity [29], existing PPI work [12], [13], [22] constructs its index by using a grouping approach. The idea is to assign the providers into disjoint privacy groups, so that true positive providers are mixed with the false positives in the same group and are made indistinguishable. Then, a group reports binary value 1 on a term t_j as long as there is at least one provider in this group who possesses the term. For example, consider terms are distributed in a raw matrix M as in Figure 2. If providers p_2 and p_3 are assigned to the same group, say g_1, then in the published PPI group $g_1$would report to have term t_0 and t_2 but not t_1, because both p_2 and p_3 do not have term t_1.

1) Privacy under primary attack

To form privacy groups, existing PPIs randomly assign providers to groups. By this means, the false positive rate resulted in the PPI varies non-deterministically. Furthermore, grouping based approach is fundamentally difficult to achieve per-term privacy degree. Because different terms share the same group assignment, even if one can tune grouping strategy (instead of doing it randomly) to meet privacy requirement for one or few terms, it would be extremely hard, if not impossible, to meet the privacy requirement for thousands of terms. For primary attack, the privacy leakage depends on the false positive rate of row at term t_j in PPI M'. This way, the grouping based PPI can at best provide a privacy level at NOGUARANTEE for primary attacks. Our experiments in Section V-A1 confirms our analysis as well.

2) Privacy under common-term attack

The grouping based PPI work may disclose the truthful term-to-provider distribution and thus the identity of common terms. We use a specific example to demonstrate this vulnerability.

Example In an extreme scenario, one common term is with 100% frequency and all other terms show up in only one provider. For group assignment, as long as there are more than two groups, the rare terms can only show up in one group. In this case, the only common term in M' is the true one in M, in spite of the grouping strategy. This allows the attacker to be able to identify the true common terms in M and mount an attack against it with 100% confidence.

Given information of term distribution, one can fully exploit the vulnerability to amount common-term attacks. And the privacy degree depends on availability of term distribution information. For certain existing PPI [22], it directly leaks the sensitive common term's frequency σ_j to providers during index construction, leading to a NOPROTECT privacy level. Other PPI work, which does not leak exact term distribution information, still suffers from data-dependent privacy protection, resulting in a NOGUARANTEE privacy level.