ϵ-PPI: Searching Information Networks with Quantitative Privacy Guarantee

Yuzhe Tang [†] Ling Liu [†] Arun Iyengar [‡]

[†]*Georgia Institute of Technology, GA, USA, Email:* {yztang@, lingliu@cc.}gatech.edu
[‡]*IBM T.J. Watson Research Center, NY, USA, Email:* aruni@us.ibm.com

Abstract

In the information sharing networks, having a privacy preserving index (or PPI) is critically important for providing efficient search on access controlled content cross distributed providers while preserving their content privacy. An understudied problem for PPI techniques is how to provide controllable privacy preservation, given the innate difference of privacy of the different content and providers. In this paper we present a configurable privacy preserving index, coined ϵ-PPI, which allows for quantitative privacy protection levels on fine-grained data units. 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 relationship 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 costly MPC computation 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 are mutually untrusted due to their autonomous nature, 2) providers have the responsibility of protecting owners' privacy and 3) it is crucial to share information between providers from an application perspective.

An example of the information network is the emerging Nationwide eHealthcare Information Network (NHIN [1] and Healthcare software CONNECT [2]), in which patients delegate their personal medical records to the hospitals where they have visited. Different hospitals may compete for customer patients and have conflicting economic interests, which renders it hard to build full trust relationship between them. Hospitals are responsible for protecting patient privacy, which is regulated by Federal laws (e.g. HiPPA [3]). On the other hand, to provide immediate and accurate medical diagnosis and treatment, it is important to have an information sharing service; for example, when a patient is sent to hospital unconsciously, such information sharing can help the doctor be able to retrieve the patients' medical history that involves multiple (untrusted) hospitals. Another example is the cross-university online course management systems (e.g. StudIP [4], Swiki [5]). Such online systems allow sharing of access-controlled materials within groups of students and teachers; while the information sharing is crucial for collaborative learning and improved learning efficiency, it may pose threat to the student privacy, which on the other hand has to be protected with compliance of Federal laws (e.g. FERPA [6]). In these untrusted networks with needs of cross-domain collaborations, it calls for a global mechanism to protect privacy of a patient or a student, while enabling effective information sharing.

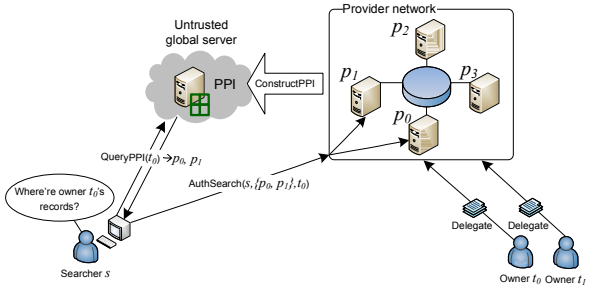


Fig. 1: The system of PPI and the provider network

To support and promote information sharing among mutually untrusted providers, privacy preserving index or PPI [7], [8], [9], [10] is proposed. The PPI aims at supporting a global search facility that can be hosted on a not-fully-trusted entity; the PPI is designed to respect providers' complete access control on personal records and the privacy. The typical working of a PPI, as will be elaborated in Section II, is a two-phase search procedure. Shown in Figure 1, a searcher, in the hope of finding certain owners' records, first contacts the PPI, which returns her a list of providers that may or

may not have the records of interests. Then the searcher then follows the list; for each provider in the list, she attempts to get authenticated and authorized before she can locally search the private records there. The hosting of a PPI is considered to be at a third-party and untrusted entity, due to difficulty of finding a global entity trusted by all (mutually untrusted) providers; for example, consider government as a candidate, which however loses trust from the public in the presence of recent PRISM scandal [11]. Therefore the PPI has to be constructed in such a way that the data owners' privacy is protected. For performance reason, existing PPI protect privacy without using encryption, but instead by adding noises to obscure the true provider-to-owner relationship.

Quantitatively Differentiating Privacy Preservation

While existing PPI's has addressed the privacy preservation, none of these approaches recognize the needs of differentiating privacy preservation for different owners and providers. To be specific, the privacy goals addressed by a PPI system is about a private fact whether "an owner t_j has his/her record stored on provider p_i". It is easy to see that disclosing the private fact regarding different owners and providers causes different levels of privacy concern. 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 their visit to a hospital, a celebrity may be more concerned about it, because even a small private matter of a celebrity can be publicized by the media (e.g., by paparazzi). To address the innate privacy difference of owners, It is therefore critical to differentiate privacy protection to address the innate different privacy concerns in a PPI system. That being said, using existing PPI approaches can not provide quantitative guarantee on the privacy preservation degree, let alone on a fine-grained per-owner basis. The cause, largely due to the degree-agnostic way of constructing PPI, is analyzed in Appendix B.

In this paper, we propose a new PPI abstraction for differentiated and quantitative privacy control, coined ϵ-PPI. Here, ϵ is a privacy aware knob that allows each owner to mark a desired privacy level when delegating data 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 exactly those providers who definitely have records of interests) and value 1 for the best privacy preservation (in this case, PPI returns all providers and a search essentially goes 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 thus vulnerable) provider from the obscured provider list when attempting to mount a meaningful attack.

Challenges: To construct the new ϵ-PPI abstraction, it poses challenges. On one hand, achieving quantitative privacy guarantee typically requires the index construction to know more about providers' data (e.g. owner distribution across providers), which entails information exchange between providers. On the other hand, providers do not trust each other, and may feel reluctant to disclose too detailed information. Therefore, it is essential to draw a clear line between what's private information and what's not during index construction, and to fully utilize the non-private information to provide as much quantitative guarantee as possible.In our proposed construction protocol, we utilize the owner frequency (i.e. the number of providers an owner has delegated her records to). Our unique insight in protocol design is that the owner frequency is private only when the value is big (i.e., when the owner's record appears almost everywhere). This is because knowing such "common" owners would give adversary confidence to make successful guess regarding any provider. By releasing and only releasing the small-value frequency, we can protect providers' privacy and deliver a quantitative guarantee.

In realizing the design protocol in a mutually untrusted network, we rely on MPC computation (i.e., secure multi-party computation [12], [13], [14], [15]) which addresses the input privacy for generic computation. However, it raises performance issues when directly applying MPC techniques in our problem setting. On one hand, current MPC computation platforms can only scale to small workload [16]; they are practically efficient only for simple computation among few parties. On the other hand, a typical PPI construction may involve thousands of owners and tens or hundreds of providers, which entails an intensive use of bit-wise MPC computation. It is therefore critical to a practical MPC protocol to efficiently carry out the computation for ϵ-PPI construction. In this regards, we propose to minimize the expensive MPC computation 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 protect providers' privacy and can tolerate collusion of up to c providers (c is configurable).

The contributions of this paper can be summarized as follows,

- We propose ϵ-PPI that differentiates the needs of privacy protection in a quantitative manner. 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 gives quantitative privacy control while enabling information sharing.
- We propose ϵ-PPI construction protocol in untrusted environment. As far as we know, this is the first PPI construction protocol without assumption on trusted party or mutual trust relationship 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-owner attack) that can break generic PPI systems. The new attack model targets on the vulnerable common owners. Our proposed ϵ-PPI is the first to resist to the common-

owner attack 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 E surveys the related work before the conclusion in Section VII.

II. Problem Formulation

A. System Model

We formally describe our system model, which involves with four entities: 1) a set of n data owners, denoted by $\{t_j\}$, each of who holds a set of personal records, 2) a provider network consisting of m providers in which a provider p_i is an autonomously operated party (e.g. a hospital or a university), 3) a global PPI server in a third-party domain, 4) data searcher who wants to find all the records of an owner of interests. There are interactions between these four entities, formulated in the following four operations.

- Delegate($< t_j, \epsilon_j >$, p_i): A data owner t_j can delegate her records to provider p_i based on her trust relationship (e.g. such trust can be built based on her visit to a hospital). Along with delegation, the owner can specify her preferred privacy 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. celebrity patient in eHealthcare network) may want to set privacy degree at a high value while an average patient may set privacy at a medium value [1]
- 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 data serving 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. For the first phase, a 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 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 with user authentication and authorization regarding searcher s; we assume each provider has already set up its local access control subsystem for authorized access. Only after authorization can the searcher search the local repository on provider p_i for records of owner t_j.

[1]To prevent every user from setting the highest value of ϵ, one possible way is to differentiate price for different privacy setting. The system has incentives to do so, since high privacy setting incurs higher costs in the provider networks.

Data model: We describe the internal data model in a PPI. Each personal record contains an owner identity t_j[2] (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 has delegated their records on provider p_i. For example, provider p_0 who has records of owner t_0 and t_1 then maintains 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 in format of a $m \times n$ matrix $M'(\cdot, \cdot)$, in which each row is for an owner, each column is for a provider and each cell is a Boolean value to indicate the membership/non-membership of the provider and the owner. For purpose of privacy preservation, there are noises or false positives 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 for the purpose of privacy preservation.

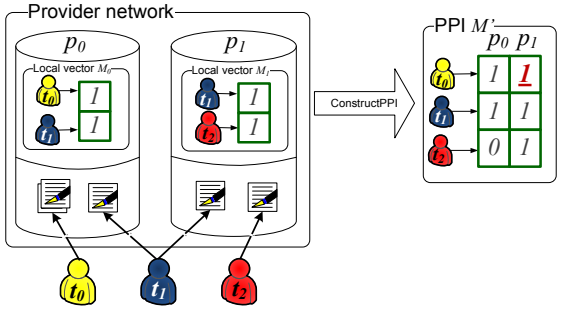


Fig. 2: ϵ-PPI model

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

TABLE I: Notations

System model symbols			
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 of constructed PPI
ϵ-PPI construction symbols			
β_j	Publishing probability of t_j	σ_j	Frequency of owner t_j
λ	Percentage of common owners	fp_j	Actual false positive rate of t_j

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

B. Threat Model

Privacy goals: In our work, we are mainly concerned with the owner-membership privacy, which is relevant to the private fact regarding which providers an owner's records belong to, that is, $M(i,j) = 1$[3]. Knowing this fact, one can infer private knowledge; for example, knowing a sport celebrity has records stored in a surgery hospital can infer that he or she may have had a medical condition possibly requiring surgery (e.g. severe injuries). For example, the fact that "the hospital of special surgery has stored medical record of Tiger Woods" could suggest that Tiger Woods . Other privacy goals related to the PPI system but not addressed in this work include searcher anonymity and content privacy. The searcher anonymity involves preventing an attacker from determining which searcher has issued queries about which owner. The content privacy involves which provider has records of what content.

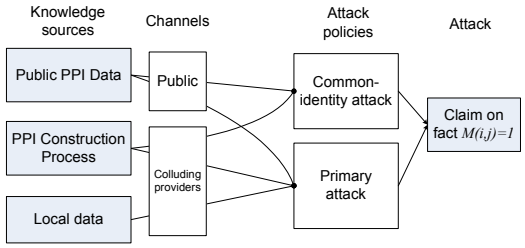


Fig. 3: An information flow model of the privacy threat

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 we consider 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 records delegated to provider p_i". To intentionally choose the attacked provider and owner, the attacker can attempt to have multiple knowledge from different channels. For example, she can learn about the public-available PPI data M', and attack only those with $M'(i,j) = 1$. Or the attacker can learn about certain parameters revealed in PPI construction process. For example, a provider exchange information with other providers for index construction, and through colluding providers, an attacker can know the exchanged information, and use it to refine the attack. Another possible source is for the attacker to learn certain provider's true vector M_i (by collusion) and use it to guess other provider's vector $M_{i'}$. Figure 3 shows the channels we consider in our attack model.
- **Common-identity attack** : In this attack, the attacker can learn about the true frequency of owner identity σ_j. Then the attacker targets his focus on those *common identities*, which appears in most of the providers. By this way, the attacker can have better chance to nail down a true positive provider and have a successful attack. For example, consider the extreme case, by learning an owner identity is with frequency $\sigma_j = 100\%$, the attacker can find a true positive provider p_i (i.e., $M(i,j) = 1$) with 100% confidence. The true frequency is revealed by existing PPI systems, either during index construction process [9] or in constructed PPI [7], [8].

In addition, this paper focuses on attacks targeting on a single owner, while a multi-owner attack boils down to multiple single-owner attacks.

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

C. Privacy Metric and Degrees

Privacy metric: We measure the privacy disclosure by the confidence an attacker can have when mounting an attacker. Formally, given an owner t_j and the knowledge of PPI data $M'(\cdot,j)$, an attacker can attack any provider, say p_i, such that $M'(i,j) = 1$. We can then measure the privacy disclosure by the probability $Pr(M(i,j) = 1 | M'(i,j) = 1)$. For all possible providers subject to attack, we consider the average probability as privacy measure. In the list of providers $M'(i,j) = 1$, the false positive rate is denoted by fp_j, and in this case the average probability is equal to the false positive rate, as follows.

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

Based on the privacy metric, we further define four privacy degrees that fully integrate with our privacy threat model. Note that in our privacy threat model, shown in Figure 3, the information flows from the source to a place where an attacker can learn, and then to a scenario where the knowledge can be exploited to form the attacker.

- **UNLEAKED**: The information can not be leaked from the source and the attacker can not know the information. This is the highest privacy protection level.
- **ϵ-PRIVATE**: The information can be leaked to attackers (through various channels). If this occurs, the PPI design protects privacy from being disclosed from the leaked information. The protection can provide a quantitative guarantee on privacy leakage. Formally, given a quantitative degree ϵ_j, we have,

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

  In particular, when $\epsilon = 0\%$, the attacker can be 100% confident about the attack, and privacy is definitely leaked.
- **NOGUARANTEE**: The information can be leaked to attackers. While a PPI design protects privacy for the case, the PPI can not provide any guarantees on the privacy protection/leakage degree. The actual privacy leakage could be unpredictable.
- **NOPROTECT**: The information is leaked from source to attack and the PPI design does not provide anything to prevent privacy leakage from the information. In this

case, privacy can be definitely leaked, corresponding to ϵ-PRIVATE with $\epsilon = 0\%$. This is the lowest level of privacy preservation, and in practice, it relies on a trusted party to protect the leaked privacy.

Based on our model, we can summarize the prior work's privacy degrees 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 [7], [8]	NoGuarantee	NoGuarantee
SS-PPI [9]	NoGuarantee	NoProtect
ϵ-PPI	ϵ-PRIVATE	ϵ-PRIVATE

D. The Degree-aware PPI Construction

We aim at achieving ϵ-PRIVATE on a per-owner basis. To be specific, this paper addresses the degree-aware PPI construction problem, formally described as following.

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 degree-aware PPI construction is how to construct a PPI to bound any search attacker's confidence, in other words, the per-owner privacy metric, under ϵ_j, with regards to all types of attacks on owner t_j described in our threat model.

III. ϵ-PPI Construction: the Computation

Our PPI construction is based on a proposed two-phase framework in which providers first collectively calculate a global value β, and then independently publishes its local vector randomly based on probability β. This framework requires complex computations and in this section, we introduce them at different granularity. First we take an overview of our two-phase construction framework, then focus on the β-calculation problem; we present the detailed computation models for β regarding different types of owner identities (i.e. common and non-common ones), and finally we perform 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, and then the second phase is to publish the private membership value regarding owner t_j and every provider p_i. We describe in detail the second phase. 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, the randomized publication follows has two rules: 1) *truthful publication*, which applies for the case of input Boolean value 1. The rule indicates that the positive membership is always truthfully published, that is, whenever $M(i,j) = 1$, $M'(i,j) = 1$. This rule guarantees that relevant providers are always in the QueryPPI result, in other words, the 100% query recall. 2) *false-positive publication*, which applies for the case of Boolean value 0. This rule indicates that the negative membership value 0 is flipped to value 1 (i.e. false positive) at probability β_j. The existence of false positive providers in the PPI can help obscuring the true owner-to-provider membership and preserves the membership privacy. We summarize the second-phase publication rules as follows. Note when Boolean value $M(i,j) = 1$, it is not allowed to publish it as $M'(i,j) = 0$.

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

For multiple owners with different β's, the second-phase publication runs independently.

An example: Consider the case in Figure 2. For owner t_0, if the β_0 is calculated to be 0.5 (the β-calculation will be elaborated soon), then provider p_1 would publish its negative membership value $M_1(0) = 0$ to value 1 at probability 0.5. In this example, it is flipped and the constructed 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 PPI contains $M'(0,2) = 0$.

B. The β Calculation Problem

In the randomized publication, β_j determines the level of false positive in the published PPI. It is critical to calculate a value of β_j so that it meets the privacy requirement regarding ϵ-PRIVATE degree. We focus on this β-calculation problem in the first phase of PPI construction.

Concretely we consider two cases: the first case is about a regular owner, the second case is about a common owner who delegates her records to almost all providers in the network. This second case should be specially taken care of, mainly because the regular computation of β_j entails the knowledge of frequency or how many providers this owner has records delegated to, and for a common owner, disclosing frequency could lead to privacy disclosure. Consider an extreme case where an owner appears in all providers (i.e., 100% frequency).

In our schemes, we differentiate the β computation for different types of owners. While for regular, non-common owners, it makes sense to set value of β, for common owners, $\beta_j = 1$. In addition, the fact that an owner is common can disclose privacy. This entails the obscuring of common owners, and we achieve that by mixing common owners with non-common ones. We propose two β-calculation schemes, respectively for non-common owners and common owners.

To construct index that meets desired privacy level, the core problem lays on the first phase, that is, how to calculate $\{\beta_j\}$ that attains the desired privacy level by $\{\epsilon_j\}$. We consider the common identities and non-common identities separately and propose different protection mechanisms.

C. Calculating β for Non-common Identity

For a single non-common identity, the negative providers suffice to meet the desired false positive rate. And we consider how to independently set β for each identity t_j in order to meet ϵ_j. Note that the second phase can be modeled as a serial of Bernoulli trials because for an identity t_j each negative provider follows the same probability $\beta(t_j)$ to flip the binary. We propose a basic policy to set β so that the expected number of flipped binaries (or false positives) can reach desired level, that is, $\epsilon_j \cdot m(1-\sigma_j)$. We have

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

The basic policy to publish with $\beta_b(t_j)$ based on expectation has poor quality in attaining the desired privacy preservation; The resulting false positive rate fp is above the configured one ϵ_j with only 50% chances, attaining success rate around 50%. Here, we define the success rate (or ratio) p_p to be the probability with which the constructed ϵ-PPI can meet the privacy requirement; in other words, the likelihood that the actual false positive rate is larger than ϵ_j. In order to address the low success ratio (i.e., 50%), we propose two new policies that intelligently increases β_b: an incremented expectation-based policy with β_d and a Chernoff bound-based policy β_c.

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,

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

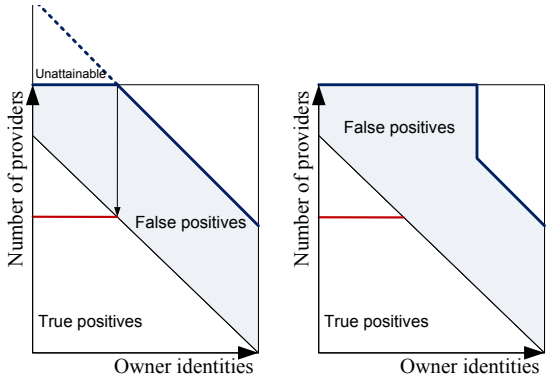
Incremental Δ can be configured by the system administrator, based on the quality requirement; the bigger the value is, the higher success rate p_p is expected to attain. However, there is no direct connection between the configured value of Δ and the success rate p_p that can be achieved, leaving it a hard task to figure out Δ based on desired p_p. This motivates us to devise a more practical policy.

Chernoff bound-based policy: Note that the randomized index publication can be modeled as a serials of Bernoulli trials. We apply the Chernoff bounds and propose an effective policy to set β. This policy allows for directly control the success rate as preferred by system administrator. Formally, it has the property described in Theorem 3.1 (with the proof in Appendix A-A).

Theorem 3.1: Given desired success rate $\gamma > 50\%$, let $G_j = \frac{\ln\frac{1}{1-\gamma}}{(1-\sigma_j)m}$ (where m is the number of providers) 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, the randomized publishing with $\beta(t_j) = \beta_c(t_j)$ statistically guarantees that the actual false positive rate in the published ϵ-PPI is larger than ϵ with success rate $p_p \geq \gamma$.



(a) Vulnerability (b) Protection

Fig. 4: Common identity vulnerability and protection

D. Calculating β for Common Identity by Mixing

Directly using the above probability setting policies is vulnerable to the common-identity attack, because the β_*[4], disclosed to (colluding) providers, allows the attacker to easily infer the original identity frequency σ (e.g., from Equation 3 one can know $\sigma_j = (1+\frac{\beta_b^{-1}}{\epsilon^{-1}-1})^{-1}$ assuming ϵ_j is publicly known), from which the common-identity attack can be formulated.

To model the common-identity vulnerability, we have a concrete example as in Figure 4a, which considers a simplified case of linear owner-to-provider distribution. As shown by the bold (and dark blue) line, the required false positives are uniform for all the identities. However, for certain identities, their negative providers are so few that the false positive is unattainable (as shown by the dotted line). And these are the common identities whose $\beta_* > 1$. The expected identity distribution of the published PPI would be as the solid bold line, which has a "plateau" area for these common identities. Observing such behavior (e.g., by mining the public PPI data), the attacker can easily identify these common identities whose desired false positive rate is unattainable, and then accordingly attacks those common identities to gain higher confidence.

Against the common-identity attack, we propose a identity-mixing technique for ϵ-PPI based on the idea of mixing common identities with certain non-common identities. Figure 4b explains our insight: by exaggerating non-common identities' β_* to 1 (i.e., all the providers claim to have the identity), we can create an augmented "plateau" area from which an attacker can not distinguish between the true common identities and the false ones (i.e., the non-common identities). Formally, for each non-common identity, we allow to exaggerate β to 1 with probability γ, that is,

$$\beta = \begin{cases} \begin{cases} \beta_*, & 1-\gamma \\ 1, & \gamma \end{cases}, & \beta_* < 1 \\ 1, & \beta_* \geq 1 \end{cases} \tag{6}$$

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

Given a set of common identities, we need to determine how many non-common identities should be chosen for mixing, in other words, how to set the value of γ. While a big value of γ protect the identity of common identities among large number of rare identities, it may unnecessarily incur high search cost. On the other hand, too small value of γ would leave common identities unprotected. In ϵ-PPI, we use the following heuristic-based policy to set γ.

- In the set of mixed identities, the percentage of rare identities should be should be at least $Y\%$. Since the number of common identity is $\sum_{\beta_* \geq 1} 1$ and the expected number of non-common identities in the set is $\sum_{\beta_* < 1} \gamma$, then we have:

$$\begin{aligned} Y\% &\leq \frac{\sum_{\beta_* < 1} \gamma}{\sum_{\beta_* \geq 1} 1 + \sum_{\beta_* < 1} \gamma} \\ \Rightarrow \gamma &\geq \frac{Y\%}{1 - Y\%} \cdot \frac{\sum_{\beta_* \geq 1} 1}{n - \sum_{\beta_* \geq 1} 1} \end{aligned} \tag{7}$$

E. β Computation: Putting It Together

We summarize the computation required by the index construction. For each identity t_j, our goal is to calculate $\beta(t_j)$ based on Equation 6 and release it to the providers so that they can publish their private local content, independently. For index construction, the computation model of β follows the information flow as below. The underline indicates the variable is private and $\Rightarrow$ indicates the complex computation involved in calculating β_*.

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

F. Privacy Analysis of Constructed ϵ-PPI

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

Privacy under primary attack: The property of the three policies to set β_* suggests that the false positive rate in the published 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's noteworthy that our ϵ-PPI is fully resistant to repeated attacks against the same identity over time, because our index is static; in other words, once getting published, it stays the same and unchanged.

Privacy under common-identity attack: For common-identity attack, the attacker's confidence in choosing a true common identity in M depends on the percentage of true common identities among the (mixed) common identities in M'. Therefore the privacy preservation degree is bounded by the percentage of false positives (i.e., the non-common identities in M published as common identities in M'), which equals $Y\%$. By properly setting $X\%$, s.t. $Y\% = \max_{\forall t_j \in \{\text{common identities}\}} \epsilon_j$, we can guarantee the per-identity ϵ-PRIVATE degree against the common-identity attack as well.

IV. ϵ-PPI Construction: Realization

Index of ϵ-PPI is constructed from an environment that lacks mutual trusts. This section describes the design and implementation of distributed and secure protocols that realizes the computation of $\{\beta_j\}$ described in the previous section.

A. Protocol 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 share the private membership vector with other. On one hand, the secure computation would requires multi-party computation (or MPC) which respects the per-party input privacy. Current techniques for MPC only support small computation workload [16]. On the other hand, the computation required in PPI construction is complex and large scaled; the computation model involves large number of identities and providers, and even for a single identity, it involves with fairly complex computation (e.g., square root and logarithm as in Equation 5). This poses great challenges to design a practical protocol for secure PPI construction.

The design of proposed protocol follows the principle of *minimizing the secure computation* with three salient features: 1) The proposed protocol separates the secure and non-secure computations by the last appearance of private variables in the computation flow. 2) We minimize the expensive secure computation by pushing the private variables up through the computation flow. In particular, instead of first carrying out complex float-point computations for raw probability β, as in Formula 8, we push such computation down through the flow and pull up the obscuring computation for private input, as in Formula 9. 3) To improve the scalability to large number of providers, we propose a domain-specific and efficient secure computation protocol for summation that can reduce the "core" of MPC part to a practical level.

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

B. The Distributed Algorithm

Following our design, we propose a two-stage algorithm for practical and privacy preserving index construction in untrusted environment. The overall algorithm is shown in Algorithm 1; Given m binary vectors $M(i, *)$ (one per a provider p_i) it outputs the $\{\beta(*)\}$ vector for all the identities. The first stage uses a parallel SecSumShare protocol that outputs c secret shares whose sum equals the sum of all the inputs. Here, c is a user-configured security parameter which indicates the maximally tolerable number of colluding providers in the network. The output c shares are individually randomized in the sense that knowing $x < c$ shares one can deduce no information at all about the sum of c shares (i.e., the sum of all the inputs). The c shares are maintained by c randomly-picked coordinators from m providers. The second

stage starts by the c coordinators performing a CountBelow computation. As shown by Algorithm 2, given c share vectors $s(0,*),\dots s(c-1,*)$ (each $s(i,*)$ is a vector of n per-owner binaries), the CountBelow algorithm computes the number of elements in the summed vector $\sum_i s(i,*)$ that are bigger than a threshold t.

TABLE III: Algorithms for ϵ-PPI Construction

Algorithm 1 compute-beta($M(0,*),\dots M(n-1,*)$)

1: $\{s(0,*),\dots s(c-1,*)\} \leftarrow$ SecSumShare($M(0),\dots M(n-1)$)
2: $\sigma'(*)$ is calculated by $\beta_* = 1$ ▷ From Equation 3, 4 or 5
3: $\sum_{\sigma\geq\sigma'} 1 \leftarrow$CountBelow$(s_0,\dots s_{c-1}, \sigma'(*)\cdot m)$
4: $\{\beta_0,\dots \beta_{m-1}\} \leftarrow \sum_{\sigma\geq\sigma'} 1$ ▷ By Equation 9

Algorithm 2 CountBelow($s(0,*),\dots s(c-1,*)$, threshold t)

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

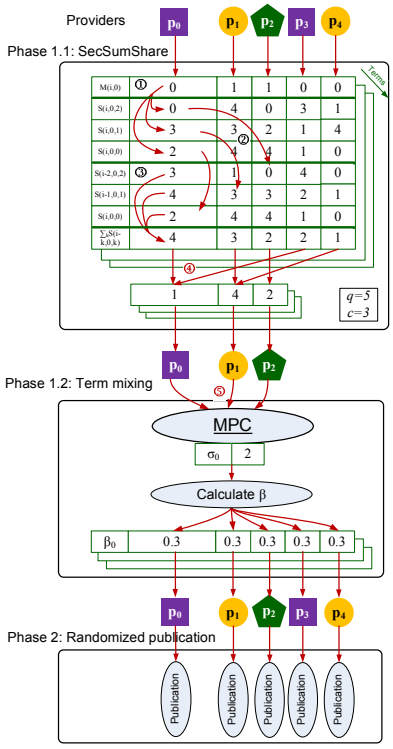


Fig. 5: Example of the index construction protocol

1) Distributed Algorithm for SecSumShare: In Figure 5, we use an example to illustrate how the algorithm of SecSumShare is executed in a distributed environment. In the example $c = 3$ and there are five providers $p_0,\dots p_4$ with one identity t_0. Out of the 5 providers, p_1 and p_2 possess the identity (i.e., $M(1,0) = M(2,0) = 1$). In the first stage, SecSumShare uses security parameter c to generate the same number of shares and performs addition on a small finite field with modulus divisor $q = 5$. It runs in the following 4 steps.

1 Generating shares: each provider p_i decomposes its local secret element $M(i,j)$ into c shares, $\{S(i,j,k)\}$, with $k \in [0, c-1]$. The first $c-1$ shares are randomly picked from interval $[0,q]$ and the last share is chosen so that the sum of all shares equals secret element $M(i,0)$ in modulo q. That is, $(\sum_{k\in[0,c]} S(i,j,k)) \mod q = M(i,j)$. In Figure 5, as depicted by arrows ①, p_0's secret element $M(0,0)$ is decomposed to 3 pieces of shares, $\{S(0,0,k)\} = \{2,3,0\}$. And $(2+3+0) \mod 5 = 0$.
2 Distributing shares: each provider p_i then distributes her 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 5, p_0 first keeps the share 2 locally, sends her second share 3 to her successor p_1 and third share 0 to 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 5, after the stage 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 her super-share to a set of c coordinators which are randomly chosen in the network. These coordinators receiving super-shares then sum the received shares up and output the summed vector $s(i,*)$ to the next-stage CountBelow protocol. In Figure 5, 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 $(4+2) \mod 5 = 1$. 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 identity t_0 is possessed by two providers. This total appearance number or identity frequency is sensitive and can not be disclosed immediately, which is why we need MPC protocol to filter out the frequency values that are too sensitive in the next stage.

2) Implementation of CountBelow computation: The secure computation of CountBelow is implemented by using generic MPC protocol. Each party corresponds to a provider in the ϵ-PPI system. Specifically, we choose a Boolean-circuit based MPC protocol FairplayMP [13] for implementation. The reason is that comparing to arithmetic-circuit based protocol, it lends itself to the computation of comparison required in Algorithm 2 (i.e., in Line 4). In particular for $c = 2$, CountBelow is essentially a comparison because the output is equal to the times one party wins (i.e., $s(0,i) > t - s(1,i)$). And the problem can be reduced to a Millionaire problem [17]. The distributed algorithm to carry out MPC (and thus our MPC-based CountBelow computation) can be found in [12], [13].

C. Privacy Analysis of Constructing ϵ-PPI

For privacy analysis of the index construction process, we mainly consider the semi-honest model which is consistent with previous MPC works [13]. The SecSumShare guarantees: 1) $(2c-3)$-secrecy of the input [9]: With less than c providers in collusion, none of any private input can be learned by providers other than its owner. 2) c-secrecy of the output: 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 security and privacy of CountBelow relies on that of the MPC used in implementation. The generic MPC technique can provide information confidentiality against colluding providers on c participating parties [13]. The output β does not carry any private information, and is safe to be released to the (potentially untrusted) providers.

We analyze the security property of the SecSumShare protocol. Basically SecSumShare is by itself a distributed secret sharing protocol; the output c shares can protect information secrecy as described in Theorem 4.1. The proof can be found in Appendix A-B.

Theorem 4.1: The SecSumShare protocol is a (c, c) secret sharing scheme of the sum of its inputs. Specifically, SecSumShare takes m ($m \gg c$) binary 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 the same to the prior distribution,

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

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

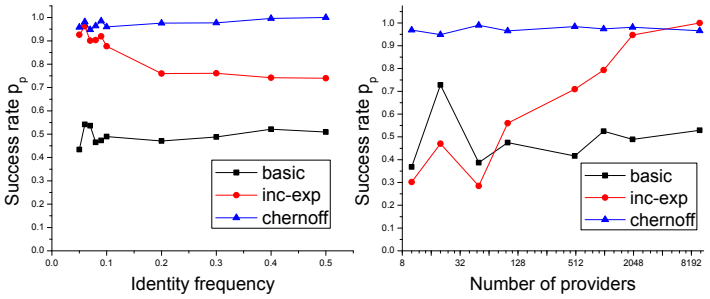
V. Experiments

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

A. Effectiveness of Privacy Preservation

Experimental setup: To simulate the information provider network, we used a distributed document dataset [18] 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 [19]. 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 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) Effectiveness of different β-calculation policies: We evaluate the effectiveness of three β-calculation policies with ϵ-PPI, and the result shows the advantages of Chernoff bound-based policy in meeting desired privacy requirement. In the experiment, we have tested various parameter setting, and show the representative result with the following values: $\Delta = 0.02$ in incremented expectation-based policy and expected success ratio $\gamma = 0.9$ in the Chernoff bound based policy. The default false positive rate is set at $\epsilon = 0.5$. The experiment results are reported in Figure 6: we consider success ratio as the metric. In Figure 6a, we vary identity frequency from near 0 to 0.5 (due to lack of identity frequency above 0.5) with the number of providers fixed at $10,000$, and in Figure 6b we vary the number of providers with fixed frequency at 0.1. It can be seen from the results that while 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 situations; the expectation-based policy is not configurable and constantly has its success ratio to be around 0.5. This is expected because 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 unsatisfactory for common identities with high frequency (as in Figure 6a) and in the relatively small network of few providers (as in Figure 6b). On the other hand, the high-level privacy preservation of Chernoff bound policy comes with reasonable extra search overhead, as demonstrated in Appendix D.



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

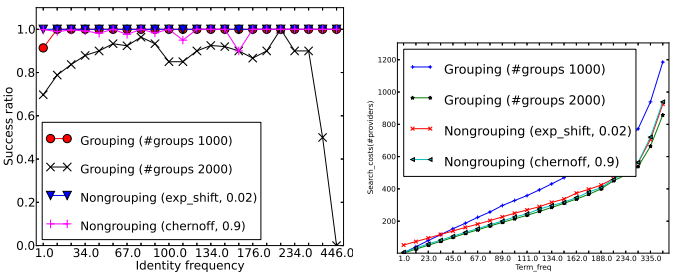
Fig. 6: Quality of privacy preservation

2) ϵ-PPI versus Existing Grouping-based PPI's: The experiments compare the ϵ-PPI with existing PPI's. The existing PPI's [7], [8], [9] are based on a grouping abstraction, that is, to organize providers 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 non-grouping approach.

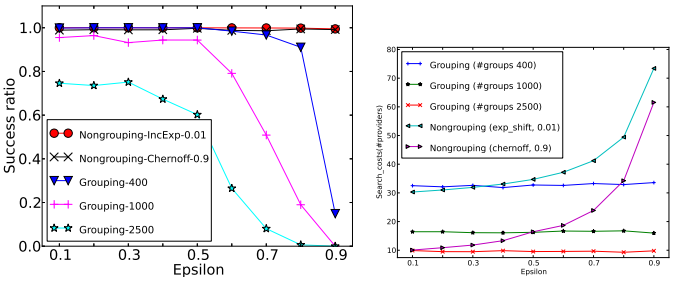
In the experiment, we measure the success ratio of privacy preservation and search performance. Grouping PPI's are tested under different group sizes. Given a network of fixed providers, we use the group number to change average group size. We test grouping PPI with the Chernoff bound-based and the incremented expectation-based policy under the default setting. Expected false positive rate is configured at 0.8 and the number of providers at $10,000$. We uniformly sample 20 times and report the averaged results.

Results are illustrated in Figure 7. Non-grouping PPI generally performs better than the grouping in terms of privacy-performance tradeoff; For the grouping PPI with comparable privacy preserving degree to the non-grouping, as exemplified by the "Grouping ($\#$groups 1000)" series in Figure 7a, it incurs unnecessarily higher search overhead than the non-grouping approach, as in Figure 7b. For grouping PPI configured to achieve similar search performance, its success ratio is much worse than that of non-grouping PPI, as shown by the "Grouping ($\#$groups 2000)" series. It can also be seen that Grouping PPI's success ratio is rather irregular and unpredictable for various identities of different frequencies. This is because with 2000 groups, sample space in each group is too small (i.e., with 50 providers) to hold a stable result in terms of privacy success ratio.



(a) Success ratio (b) Search cost

Fig. 7: Comparing non-grouping and grouping: varying identity frequency

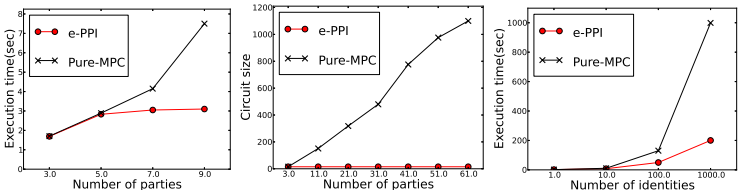


(a) Success ratio (b) Search cost

Fig. 8: Comparing the non-grouping and grouping: varying ϵ

We further show the result with changing false positive rate ϵ, that demonstrates PPI performs equally well for both sensitive identities and non-sensitive identities. In Figure 8 we report the success ratio and search costs under different false positive rates. It can be clearly seen that the non-grouping PPI treats privacy preservation as first-class citizen in its design, as opposed to grouping PPI. In particular, when ϵ grows big in Figure 8a, the success ratio of grouping PPI's quickly degrades to 0, rendering its privacy quality unacceptable. This stems from the grouping PPI's design that treats all the identities the same way. This set of experiments shows that the privacy degree of non-grouping PPI's can be effectively tuned in practice, implying the ease of configuration and more control exposed to applications.

B. Performance of Index Construction



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

Fig. 9: Performance of index construction protocol

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 [13] which is based on Boolean circuit. 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 third-party library Netty [20] for network communication and Google's protocol buffer [21] for object serialization. We conduct experiments on a number of machines in Emulab [22], [23], each equipped with a 2.4 GHz 64-bit Quad Core Xeon processor and 12 GB RAM. In the experiment, different number of machines is tested, 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 reduced-MPC approach as in ϵ-PPI construction protocol against a pure MPC approach. The pure MPC approach does not make use of SecSumShare protocol to reduce the number of parties involved 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 9a. It can be seen that the pure MPC approach generally incurs longer execution time than our reduced-MPC 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 reduced-MPC approach increases slowly. This difference is due to that the MPC computation in our reduced-MPC approach is fixed to c parties and does not change as number of providers m grows. And the parallel SecSumShare in reduced-MPC 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 scalability study 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[5] in 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 number of party involved. At last, we also study the scalability to running the protocol with multiple identities in a three-party network. The result in Figure 9c shows that ϵ-PPI construction grows with number of identities at much slower rate than that of pure MPC approach.

VI. Related Work

This section surveys related work on indexing support on untrusted servers. We focus on information confidentiality or privacy on secure index design, and do not survey the issues of integrity and authenticity.

Non-encryption based privacy preserving index: 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 [24], existing PPI work [7], [8], [9] 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 index, many existing approaches [7], [8], [25] assume providers are willing to disclose their private local indexes, an unrealistic assumption in a network lack of mutual trusts between providers. SS-PPI [9] 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'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.

Secure index and search-able encryption: Building search-able indexes over encrypted data has been widely studied in the context of both symmetric key cryptography [26] and public key cryptography [27], [28], [29]. In this architecture, content providers build their local indices and encrypt all the data and index, before submitting them to the untrusted server. During query time, the searcher first gets authenticated and authorized by the corresponding content provider, then contacts the untrusted server and searches against the encrypted index. This system architecture makes 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.

[5]Regarding the detailed definition of circuit size and the exact correlation between circuit size and execution time, it can be found in FairplayMP [13].

VII. Conclusion

In this paper, we propose ϵ-PPI for quantitative privacy control in information networks. The privacy of our ϵ-PPI can be controlled by each individual in a quantitative fashion. We identify a vulnerability of generic PPI on protecting common owner identities, and address this vulnerability in our ϵ-PPI design by proposing an identity mixing technique. We have implemented the index construction protocol without any trusted party and applied a performance-optimization design that minimizes the amount of secure computation. We built a generic privacy threat model and performed security analysis which shows the advantage of ϵ-PPI over other PPI system in terms of privacy preservation quality.

References

[1] "Nhin: http://www.hhs.gov/healthit/healthnetwork."
[2] "Connect, http://www.connectopensource.org/."
[3] "Hippa, http://www.cms.hhs.gov/hipaageninfo/."
[4] "Studip, http://www.studip.de/."
[5] "Swiki, http://en.wikipedia.org/wiki/swiki."
[6] "Ferpa, http://www2.ed.gov/ferpa."
[7] M. Bawa, R. J. B. Jr., and R. Agrawal, "Privacy-preserving indexing of documents on the network," in *VLDB*, 2003, pp. 922–933.
[8] 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.
[9] L. L. Yuzhe Tang, Ting Wang, "Privacy preserving indexing for ehealth information networks," in *CIKM*, 2011, pp. 905–914.
[10] 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.
[11] "Prism, http://en.wikipedia.org/wiki/prism_(surveillance_program)."
[12] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay - secure two-party computation system," in *USENIX Security Symposium*, 2004, pp. 287–302.
[13] 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.
[14] W. Henecka, S. Kögl, A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, "Tasty: tool for automating secure two-party computations," in *ACM Conference on Computer and Communications Security*, 2010, pp. 451–462.
[15] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen, "Asynchronous multiparty computation: Theory and implementation," in *Public Key Cryptography*, 2009, pp. 160–179.
[16] A. Narayan and A. Haeberlen, "DJoin: differentially private join queries over distributed databases," in *Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI'12)*, Oct. 2012.
[17] A. C.-C. Yao, "Protocols for secure computations (extended abstract)," in *FOCS*, 1982, pp. 160–164.

[18] J. Lu and J. P. Callan, "Content-based retrieval in hybrid peer-to-peer networks," in *CIKM*, 2003, pp. 199–206.
[19] D. Hawking, "Overview of the trec-9 web track," in *TREC*, 2000.
[20] "Netty: http://netty.io."
[21] "Protobuf: http://code.google.com/p/protobuf/."
[22] "http://www.emulab.net/."
[23] 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.
[24] 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.
[25] 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.
[26] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE Symposium on Security and Privacy*, 2000, pp. 44–55.
[27] 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.
[28] 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.
[29] 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.
[30] M. Mitzenmacher and E. Upfal, *Probability and computing - randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.
[31] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele, "Secure computation on floating point numbers," in *NDSS*, 2013.
[32] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *USENIX Security Symposium*, 2011.
[33] L. Kissner and D. Song, "Privacy-preserving set operations," in *in Advances in Cryptology - CRYPTO 2005, LNCS*. Springer, 2005, pp. 241–257.
[34] M. Wright, M. Adler, B. N. Levine, and C. Shields, "An analysis of the degradation of anonymous protocols," in *NDSS*, 2002.
[35] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, 1998.
[36] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, "Anonymous connections and onion routing," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 1997, pp. 44–54.
[37] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *J. Cryptology*, vol. 1, no. 1, pp. 65–75, 1988.
[38] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.
[39] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," in *KDD*, L. Getoor, T. E. Senator, P. Domingos, and C. Faloutsos, Eds. ACM, 2003, pp. 505–510.

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, $Pr(X \leq (1-\delta)E(X)) \leq e^{-\delta^2 E(X)/2}$ [30], 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,

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

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

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

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 directly follows the fact that $\sum_{\forall ii \in [0,c-1]} s_{ii} = \sum_{\forall i \in [0,m-1]} v_i$.[6]

To prove secrecy, we start by examining the relationship between each output value and input shares. It is easy to see that the SecSumShare protocol uses a (c,c) secret sharing to split each private input v_i. The generated c shares for each input value is distributed to a distinct one output value s. In particular, for $v = M(i,j)$, its k-th share $S(i,j,k)$ is included in the output value $s_{(i+k)\%c}$. By this mean, each output value has one and only one share for each private input v_i. Therefore, when an adversary knows at most $c-1$ output values, at least one share of each private input is still unknown to him. This leaves the value of any input completely undetermined to this adversary, thus the sum of input values completely undetermined. ■

[6]This fact can be simply supported by the intuition that each line in Figure 5 preserves the sum of the numbers in this line. The formal proof of this property can be found in [9].

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 [24], existing PPI work [7], [8], [9] 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.

a) 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-A2 confirms our analysis as well.

b) 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 [9], 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.

Appendix C
Resistance against More Attacks

A. Resistance against Background Knowledge Exploit

Background knowledge attack : The attacker may already have background knowledge about the private fact $M(i,j) = 1$. And the attacker exploits this background knowledge in association with other attacks to gain higher confidence. Since we can not prevent the attacker using her background knowledge, we clarify our design goal for ϵ-PPI is to not amplify the attacker's certainty. We discuss this attack in Appendix C (due to space limit).

We consider the attacker can have background knowledge regarding $M(i,j) = 1$. Suppose the background knowledge itself can lead to a confidence $Pr'(i,j)$. Given the background knowledge and knowledge learned from the channels of our privacy model with confidence $Pr(i,j)$, the attacker need to make a decision on which knowledge to exploit for mounting an attack.

- **Probability-based policy**: The attacker chooses to believe whichever knowledge with the higher probability, that is, the confidence of attacker's final choice $Pr_f(i,j) = \max(Pr(i,j), Pr'(i,j))$. Given p_i and t_j, the attacker always claim provider p_i does have term t_j.
- **Uncertainty-based policy**: The attacker chooses to believe the knowledge with more certainty. That is, Pr_f is chosen among Pr and Pr' such that:

  $$\|Pr_f(i,j) - \frac{1}{2}\| = \max(\|Pr(i,j) - \frac{1}{2}\|, \|Pr'(i,j) - \frac{1}{2}\|)$$

  Based on Pr_f, the attacker makes a guess about $M(i,j)$ accordingly. If $Pr_f \geq \frac{1}{2}$, the attacker claims *positively* that provider p_i does have term t_j. Otherwise, it claims the opposite. Comparing to the probability-based policy, this policy allows an attacker to have the choice of making guesses accordingly.

Different policies lead to different attacks. For example, suppose the background knowledge can lead to attacker's confidence of $Pr' = 0.1$, and the knowledge from our privacy model leads to attacker's confidence of $Pr = 0.6$. The probability-based policy would favor the PPI's knowledge(since $Pr > Pr'$), while uncertainty policy favors the background knowledge.

B. Resistance against Collusion of Accessible Providers

In the primary attack, we consider the attacker's ability to verify certain PPI result with the help of his/her accessible providers. With this ability, an attacker can see three types of providers in a PPI result of term t_j: 1) true positive

providers, who possess data records of t_j which is accessible to the attacker; 2) negative providers, who is accessible and don't have records of term t_j; 3) inaccessible providers who don't have any records of term t_j that is accessible to the attacker. In this case, false positive providers could be either negative providers or inaccessible ones. This allows the PPI model to achieve a higher false positive rate in the case of verified results. For instance, consider a PPI search that returns 5 providers, out of which 2 are false positives. By checking each provider in the result list, the searcher could distinguish the true positives $\{p_0, p_2\}$, the negatives $\{p_1, p_3\}$ and the inaccessible p_4, as shown in Table IV. Given positive inaccessible providers, the attacker can not distinguish them the case that they truly don't possess the term from the case that they possess the term but the record is not accessible. In the end, the authorized searcher still can not identify the false positive among $\{p_1, p_3, p_4\}$, which leads to false positive rate $\frac{2}{3} > \frac{2}{5}$.

TABLE IV: Example of the primary attack with verified results: it shows the check results from providers, in which 1 represents the confirmation of possession of the searched term, 0 denial of possession, and ? no information returned.

	p_0	p_1	p_2	p_3	p_4
Check result	1	0	1	0	?

Appendix D
Extra experiments on search costs

A. Effect of β-calculation policies

From Figure 10, the search costs of Chernoff bound policy are no more than 105% of those of expectation-based policy. In short, the Chernoff bound-based policy achieves all-around privacy preservation in high quality, yet with low extra search costs.

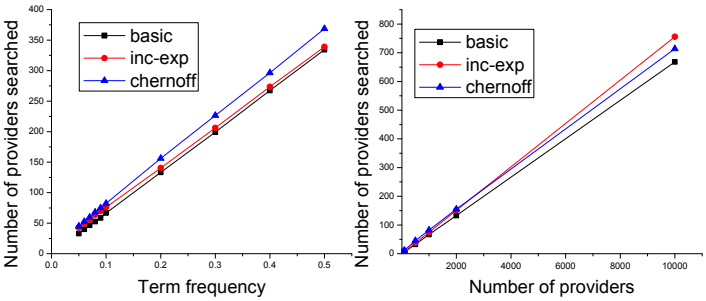


Fig. 10: Extra search costs

Appendix E
Extra Related Work on Privacy Preserving Techniques

In this section, we review several privacy preserving techniques that are applicable to constructing ϵ-PPI.

Practical MPC: Index construction of ϵ-PPI is closely related to secure multi-party computation (or MPC). Recently emerges a body of research work [12], [13], [14], [15], [31] dedicated towards practical MPC. Traditional MPC work largely follows two models, the garbled functions used for Boolean circuits and the homomorphic encryption used for arithmetic calculation. Towards efficient and practical MPC systems, Fairplay [12], [13] implements computation of Boolean circuits for two or more parties, and VIFF [15] is a runtime for computation of arithmetic circuits. MightBeEvil [32] supports efficient two-party computation via garbled circuit. Based on the observation that different computation models can lead to different performance for a given workload, TASTY [14] proposes to use a modular design which divides the whole workload into several modules, each of which is mapped to a specific MPC model. The modular computation is realized by a scheme that can convert the encrypted or garbled data between modules. Recent work [31] extends the domain of practical MPC (from integers) to floating point numbers by using Shamir's secret sharing. Due to the inefficiency of using generic MPC for large scale workload, DJoin [16] proposes to use an efficient but domain-specific primitive for set operations [33] in combination with generic MPC to perform private join computation. Our ϵ-PPI construction protocol reduces the generic MPC part by using an efficient secure sum protocol.

Anonymous communication protocols: Anonymous communication protocols have drawn a great deal of attentions for the decades. Anonymous protocols on a continuous network connection aim at hiding the sender's identity from the overt receiver. A common feature of all proposed protocols for anonymity is the selection of a group of nodes that work together to send messages [34], [35], [36], [37]. In particular, DC-Net [37] executes a 2-stage protocol to perform a secure computation of the Boolean-OR function. DC-net first sets up pair-wise shared Boolean keys between all participants, and requires each participant to release the parity of all keys shared with him/her and his/her Boolean private input. The final results are the total parity of all released values. DC-net is secure against the colluding attacks: its anonymity is breached only when all of the nodes who share keys with the owner of the private input collude to reveal their Boolean keys among themselves. Our SecSumShare protocol bears resemblance to the DC-net, but it extends the DC-net from the Boolean domain to field $\mathbb{Z}_c$ and performs without all-to-all communications. This difference makes SecSumShare protocol scalable in large networks.

Randomized responses: Randomized Response (RR) techniques [38] were proposed and developed in the statistics community for protecting surveyee's privacy. Randomized response is required to correctly estimate the percentage of people in a crowd that have certain attribute, say A. When answering the survey question, surveyees are allowed to tell a lie with a predefined probability β. When it returns σ' percent positive answers (i.e., claiming they have attribute A), the true percentage, denoted by σ, can be deduced from the following equation,

$$\sigma' = \sigma \times (1 - \beta) + (1 - \sigma) \times \beta \tag{13}$$

Although the interviewer learns the answers from surveyees, he/she does not know weather or not the surveyee lies. Thus the respondents' privacy is preserved. Randomized responses have recently been used in privacy preserving data mining [39]. Our randomized publishing protocol adapts the randomized responses to the privacy preserving indexing scenario with non-trivial changes. Specifically, the original randomized responses equally treat two cheating cases, (i.e., $0 \rightarrow 1$ and $1 \rightarrow 0$). This does not apply in privacy preserving index, since the two cheating cases have different implications here: publishing index possession from 0 to 1 means preserving content privacy with extra search overhead, while that from 1 to 0 implies hurting search result recall (yet with no impact on privacy preservation). Therefore, our proposed randomized publishing protocol differentiates the two cheating cases, by disabling $1 \rightarrow 0$.